Image-set, Temporal and Spatiotemporal Representations of Videos for Recognizing, Localizing and Quantifying Actions

by

Xiang Xiang

A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy.

Baltimore, Maryland

June, 2018

Abstract

This dissertation addresses the problem of learning video representations, which is defined here as transforming the video so that its essential structure is made more visible or accessible for action recognition and quantification. In the literature, a video can be represented by a set of images, by modeling motion or temporal dynamics, and by a 3D graph with pixels as nodes. This dissertation contributes in proposing a set of models to localize, track, segment, recognize and assess actions such as (1) image set via aggregating subset features given by regularizing normalized CNNs, (2) image set via inter-frame principal recovery and sparsely coding residual actions, (3) temporally local models with spatially global motion estimated by robust feature matching and local motion estimated by action detection with motion model added, (4) spatiotemporal models 3D graph & 3D CNN to model time as a space dimension, (5) supervised hashing by jointly learning embedding and quantization, respectively.

State-of-the-art performances are achieved for tasks such as quantifying facial pain and human diving. Primary conclusions of this dissertation are categorized as follows: (i) Image set can capture facial actions that are about collective representation;

ABSTRACT

(ii) Sparse and low-rank representations can have the expression, identity and pose cues untangled and can be learned via an image-set model and also a linear model;
(iii) Norm is related with recognizability; similarity metrics and loss functions matter;
(v) Combining the MIL based boosting tracker with the Particle Filter motion model induces a good trade-off between the appearance similarity and motion consistency;
(iv) Segmenting object locally makes it amenable to assign shape priors; it is feasible to learn knowledge such as shape priors online from Web data with weak supervision;
(v) It works locally in both space and time to represent videos as 3D graphs; 3D CNNs work effectively when inputted with temporally meaningful clips;
(vi) the rich labeled images or videos help to learn better hash functions after learning binary embedded codes than the random projections.

In addition, models proposed for videos can be adapted to other sequential images such as volumetric medical images which are not included in this dissertation.

Primary Readers: Gregory D. Hager and Trac D. Tran.

Secondary Readers: Stefano Soatto, Austin Reiter, Alan L. Yuille, and Rene Vidal.

Acknowledgments

Thanks to my parents, my family, all my advisors, mentors and coworkers along this journey. My wife has accompanied me along the journey since the early period when she was still my girlfriend. Later on we got married and then had a daughter. A lot of progress happened since the birth of my daughter. The joy brought by the little cute baby makes me have courage to overcome any obstacle until this point.

At Johns Hopkins, I appreciate the long-term supervisions from Trac D. Tran (since 2014) and Gregory D. Hager (since 2012), the prompt advise on this dissertation from Gregory D. Hager, stagewise mentoring from Alan L. Yuille, Austin Reiter, Daniel Mirota, Minh Dao and Sang P. Chin, fruitful collaborations with Feng Wang, Ye Tian, Fabian Nino and Hao Jiang, occasional discussions with Rene Vidal, Raman Arora, Russell H. Taylor, Simon Leonard, Kevin Duh, Nassir Navab and many others, helpful conversations with the lab members of Trac, Greg, Alan, Rene and LCSR, classmates in CS, ECE and AMS, and many schoolmates met in various activities. I also appreciate Daniel Robinson and Mark Dredze for serving in my GBO committee.

It has been 10 years since I get into the field of computer vision. Deepest ap-

preciation to Wuhan University for offering courses like Digital Image Processing, Pattern Recognition and Embedded System Design, which I took in the fall of 2007. However, at that time those are still just ordinary subjects in my course list. Pattern Recognition was taught by Li Li who became one of the team mentor of the Wuhan University Team attending the 2008 Intel Cup Embedded System Contest. In the spring of 2008, Yingbo Xie was the other team mentor who selected me into the team. Together with Wenhui Chen and Du Zeng, we designed an object tracking camera with advices from the lab members of Dexiang De. I seriously stepped into the literature of computer vision through this project. Before that, I tried to develop my interest in database system and data mining, and was actively involved in student activities such as serving in the chairman board of the student union. Since then, I started to focus on computer vision and the related. First, I applied for an exchange program in my senior year at Osaka University where I stayed in Yasushi Yagi's lab for moving object segmentation and applied for graduate studies at Chinese Academy of Sciences where I was advised by Xilin Chen for video coding and object tracking. From there, I formally started my journey in computer vision. I also appreciate Hong Chang and Jiebo Luo for refining the writing of my first paper in major vision conferences. Till today, this dissertation is about computer vision and particular video analytics. I want to give them all deep appreciation for guiding me into the field and thank the lab members of Yagi, VIPL, Ichiro Sakuma and a few others. Particularly many thanks to Hao Li, Wei Zheng, Zhe Li, Lam Tran, Wentao Zhu and Hao Zhu.

ACKNOWLEDGMENTS

Life is getting better while feelings are getting more blurry. My wife's grandfather passed away at the age of 96 yet very suddenly earlier this year. My own grandfather passed away at the age of 76 three years ago during my summer internships at California. I flied back twice and fully realized that I could hardly make career progress without a stable family. Actually in recent years when I am abroad, I gradually saw people in my grandfather's age pass away one after another. The feelings are subtle in terms of facts that I still remember them here and there while I have not stayed side by side with them for a while. Sometimes, I feel that it is so ridiculous for me to spend the whole night comparing which job offer is better while I hardly spend the time to chat with the old generations. I think that I will definitely regret when it gets to the point that there are few people in the old generation. I feel it is suffering to stay abroad for so long time. How can I balance my just normal life and the good old days? In the job offer letters, certain days of paid vacations are listed very clearly. Those are quite short compared to the vacation in school days. I would imagine that I will spend even shorter time hanging out together with them. However, I do appreciate the career advices given by Chengjie Tu, Justin Lorts and Chenyang Xu.

This dissertation work was made possible, in part, by the generous support of the fellowship from the China Scholarship Council, the teaching assistantship from the Johns Hopkins University's Department of Computer Science, and research grants from National Institute of Health and National Science Foundation.

Dedication

Figure 0.1: This dissertation is dedicated to my parents. Picture taken in 1989.

Contents

List of Tables

LIST OF TABLES

List of Figures

Chapter 1

Introduction

To begin with, this chapter first describes the problems and challenges that are either thoroughly addressed with or briefly touched upon in this dissertation. Then, contributions are listed. Afterward, this chapter gives video analytics applications that partially provide the problems to be solved and motivate the models developed later in this dissertation. In the end, the outline of this dissertation is given.

1.1 Problems and challenges

Videos frequently appear in today's life. The signal processing community views videos as multi-channel signals. As a result, tasks include denoising, compression, coding, communication and so on. Differently, the computer vision community views videos as a data modality to store and process visual information. As a result, the

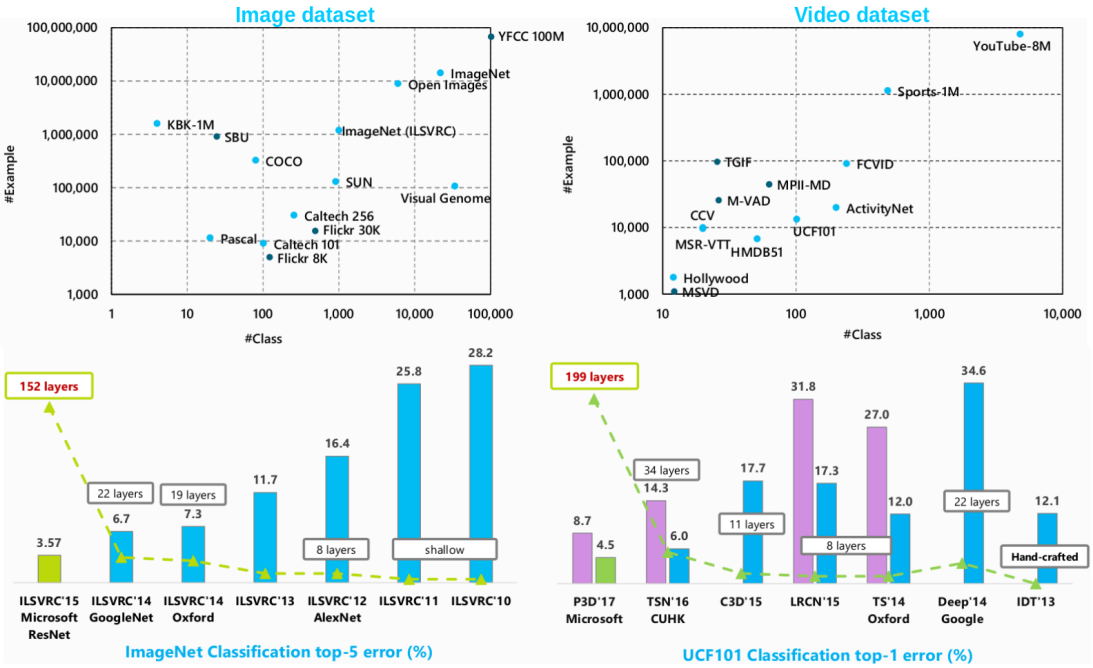


Figure 1.1: Progress of image and video recognition by 2017. Credit: Tao Mei.

tasks are more about inferring the appearance, motion, geometry, and semantics than manipulating the data modality itself.

Indeed, computer vision is hard from all aspects including high-level tasks such as recognizing an action or an event, mid-level tasks such as tracking, segmenting and classifying an object or region in dynamic scenes [10–12] and low-level tasks such as robust feature matching on non-planar texture-less surfaces. In the following, technical problems are given.

As shown in Fig. 1.1, the trend on building image recognition datasets reoccurs on building video datasets. However, the stage-wise progress of the video recognition tasks is always 2 to 3 years later than that of the image recognition tasks.

1.1.1 Action recognition, quality assessment, and video classification

For example, facial expression recognition, facial action intensity estimation, human action recognition, human action quality assessment, video face recognition, video categorization. Compared with images, videos bring in an extra dimension - the temporal dimension. Similar to images, computer vision tasks on videos also have a coarse-to-fine structure. Video classification is just as image classification: given a video, predict the category. For example, if the dataset contains various videos collected from ©YouTube, then predict the topic categories which might not be unique, say, 'entertainment', 'indoor' as keywords.

Object detection in videos is roughly equivalent to the so-called action detection - detecting where (*i.e.*, spatial object localization) and when (*i.e.*, temporal localization or called temporal segmentation) there exist object actions (say, human actions, facial actions). The task can be even more in detail than a bounding box - action segmentation in the pixel level which gives the object silhouettes over time. Some other semantic tasks in videos such as group activity recognition tend out to be easier than doing that in a single image. The temporal information can rule out some ambiguities. Some tasks are unique for videos such as the temporal event or action segmentation, event detection and change detection which is also called abnormal event detection or equivalently anomaly detection.

Figure 1.2: Characteristics of facial actions in the wild under various poses.

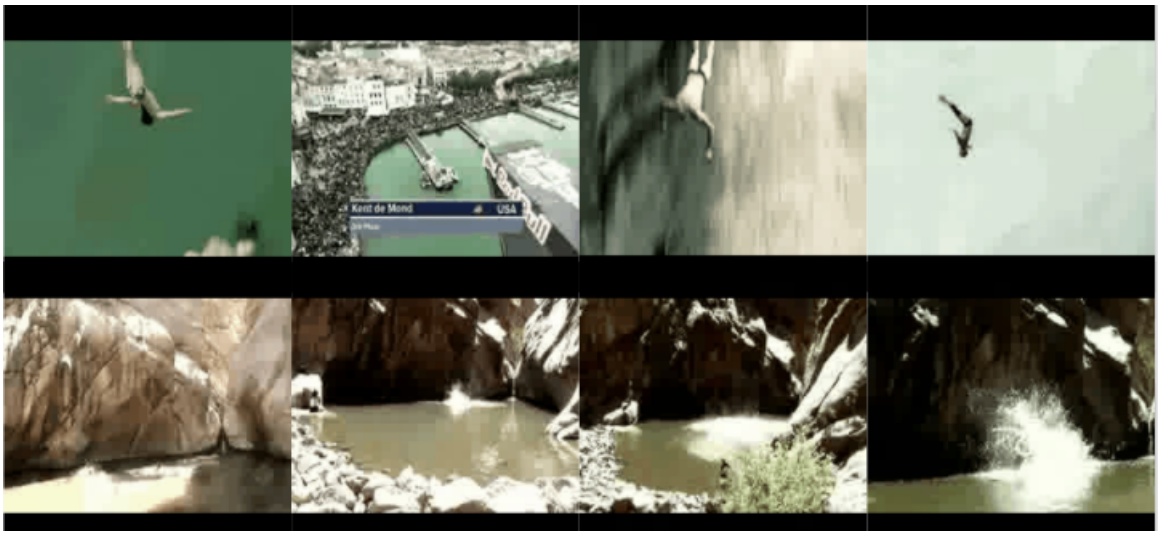


Figure 1.3: The variation of a human action all named 'diving'. The first row is platform diving while the second row is cliff diving.

1.1.2 Event detection and temporal action localization/segmentation

For example, facial expression detection, change detection, video summarization. In a number of surveillance applications, anomaly detection, or called abnormal event detection, is a highly demanded problem to be solved. This dissertation does not directly address the problem of localizing an event temporally. As for summarizing a video, the problem is often posed in terms of keyframe selection and namely subset selection [13–18]. An hours-long video of certain events such as a soccer game can be summarized using a single headline photograph which might be the celebrating players or winning fans, a set of photos taken during the process of the game, a minutes-long recap highlight video, and even a text or oral broadcasting through web pages or radios. Just to give a sense of the models described in the following, intuitively an image-set collaborative representation is the set of photos which all together collectively describe the games while the order of the photos turns out to be normally random. The temporal representation is very much in the sense of a text or oral broadcasting of the game. The game may not be watched in person but the game process is described accurately. Furthermore, the spatiotemporal representation is similar to the highlight videos which captures the scene just as photos but also describes the rough game process.

1.1.3 Spatial action localization/segmentation

For example, facial action unit detection, camera motion estimation, action detection, moving object detection, tracking and segmentation. Instead, spatially localizing an event or action is one of the problems that have been examined by this dissertation. Generally speaking, the action localization task is similar to object detection: the goal is to spatially/temporally localize a recognized action within a video, often using a per-frame bounding box representation [19]. When there contain noisy frames without actions, the spatial and particularly temporal localization is also sometimes called event detection [20], action discovery [21] or video object discovery [22]. However, in this dissertation, it is assumed that the action is consecutively happening. Namely, it is sure that there is a moving object in each frame. Moreover, challenges in the wild shown in Fig. 1.3 and Fig. 1.2 make those problems hard to solve.

1.1.4 Video retrieval

For example, video search, indexing, and ranking. Once a video's topic category or action category are known, it enables the content-based video retrieval, for example, video hashing. Activity search in video means that given a target activity, a system automatically detects and temporally localizes all instances of the activity.

1.2 Contributions

This dissertation addresses the problem of learning video representations for action recognition and quantification. It contributes in proposing a set of models to localize, segment, recognize and assess actions such as the simply image-set models. Based on linear signal representations, it proposes inter-frame principal recovery and sparsely coding residual actions, which performs the best on facial expression recognition among sparse coding methods and not far away from CNN.

Based on CNN yet for videos, it proposes image-set models via aggregating subset features given by regularizing normalized CNNs. It proves the Softmax loss bound after normalization as well as proves that Softmax loss is scale-invariant. It develops the first layer specialized in normalizing both features and weights in the literature to boost the performance of face verification and sets the new state-of-the-art performance based on the previous state-of-the-art. It also develops the first head-pose-based K-means layer for deep face networks, becomes the first time to apply the regularizing loss idea to pain assessment, pushes the performance by a large margin, sets the new state-of-the-art, and also proposes a more sensible evaluation metric.

Moreover, it exploits temporal dynamics in videos and proposes new temporal motion models. It proves the sample amount bound for RANSAC to success in multi-model motion estimation as well as empirically estimates the global motion by feature matching and validates its uncertainty. It estimates the local motion via detecting actions by appearance yet with motion models added, which consistently boosts the

performance of appearance-based tracking-by-detection models and achieves the best overall performance that is better than the state-of-the-art TLD tracker.

Furthermore, it represents videos as 3D pixel-graphs to model time as a space dimension. Based on Graphs cuts, 3D CNNs and TCN, it proposes new spatiotemporal models to spatially segment actions and assess their qualities using 3D Graph cuts and segmental 3D CNNs, respectively. In terms of spatial action segmentation, it performs competitively with the state-of-the-art that requires offline modeling based on pre-selected templates and a pre-trained person detector. In terms of action quality assessment, it becomes the first time to score diving stage by stage in the diving skill assessment literature and performing way better than the previous state-of-the-art.

In addition, it studies the reverse problem of action recognition - action retrieval. It proposes a supervised hashing model by jointly learning embedding and quantization that is well-performed.

In summary, this dissertation has a broad coverage of the video analytics area and can provide a guideline for new researchers into this area such as those work on image recognition. For specific topics in the area, it either pushes the state-of-the-art or proposes an explainable model that other researchers can build upon or bring the idea. It attacks problems from a practical perspective but it does provide analysis, studies, hypothesis testing and even theoretical analysis for certain models. For the area in general, many of the proposals are model components that can be generalized such as the normalization layer, regularizing loss, adding motion model,

and segment-based modeling.

The state-of-the-art performances have been achieved for tasks such as quantifying actions of the facial pain and human diving. The primary conclusions of this dissertation are categorized as follows.

(1) It is verified through several proposed models that image set works effectively for facial actions that are more about summarization. The collective representation are given by (i) collaborative modeling and efficiency is given by (ii) selective aggregation.

(2) It is shown that weakly supervised online manner does not necessarily result in better robustness, but does enable recovering from the error. In addition, it is a good choice to maintain an offline classifier together with the on-line classifier. Particle filter tracker cannot track the target very accurately but does not drift far away.

However, spatiotemporal representations are needed to model human actions which are more complex both in space and time.

(3) A set of models are proposed to localize, track, segment and recognize an action . Unlike two streams, motion cues are used in a unified framework such as (a) 2D spatial and 1D temporal factorized 3D CNN, (b) path optimization on a 3D graph and (c) naturally strengthening appearance based tracker with a motion model by equating the sample in particle filtering with the sample in multiple instance learning.

These topics related with the above mentioned models have been examined more or less before the rise of deep learning. For example, many action recognition papers have been published before 2010, supported by the video datasets such as the KTH

action dataset, which is a toy dataset in today's view. In the recent 3 years or so, there are many papers that propose video recognition models. Around 2015, Recurrent Neural Network (RNN) steps on the stage with Long Short-Term Memory (LSTM) networks become the star. Fig. 1.1 tells this phenomenon a little bit. Exactly in 2015, Convolutional 3D (C3D) networks and Long-term Recurrent Networks (LRCN) are published, Nowadays, those models ideas become standard in the video anlytics area.

While a few test-of-time models come out, this area is not well organized. This dissertation comes in a timely manner. There is no way to touch upon every aspects and stay aware of every recent paper. However, the test-of-time models are examined. In the current stage of computer vision research, it is challenging to propose models from scratch. A analysis by practice methodolgy is used during the development of this dissertation. In some cases, this dissertation proposes a change, which can be a revision, a correction or a refinement, to the test-of-time models. For example, the Normalized Face (NormFace) network adds normalization layers to the existing CNNs with the Center Loss for face verification. The Segmental pseudo 3D (S3D) makes change based on the Pseudo 3D (P3D) networks, which is model futher changed from the C3D model for action recognition. This type of contribution is called incremental contributions by some reviewers. However, these changes are also favored in practice. This dissertation elaborates on those changes to make them reasonable and explainable. Some phenomena, such as the fact that the performance boost when normalizing features and weights, are found during the execution of the test-of-time

models and then are analyzed and well explained.

In some other cases, non-deep-learning models are also re-visited when they play the roles well. For example, linear signal representation, such as sparse representation, works well for coding facial actions. The boosting based trackers have been extensively examined in the literature. It has become a solid component of robust trackers that are reliably used in practice. As a result, this dissertation tends to avoid reinventing the wheels. Instead, it tries to maximize the capabilities of exisiting models. For example, the boosting based trackers always drift in the end. Other than that, it performs much better than the Kalman filter, Particle filter and other Bayesian filtering type of trackers. Similarly, Kalman filtering has become a solid tool in engineering including aerospace where reliability is highly required. But that does not mean it generalizes to every case in practice. The fact that it sometimes fails does not mean it has no value. The Bayesian filtering has been given the values for object tracking long before the development of the boosting based trackers. Based on this research philosophy, this dissertation proposes a tracker which combines the merits of both boosting and Bayesian filtering. This type of work is called $A + B$ or combination work by some reviewers. Once again, the combination way is favored in practice. This dissertation does not avoid combining models just for the sake of elegance. Instead, it combines models not for the sake of combination itself. Recently there are spatiotempiral regional proposal works published. It naturally combines appearance-based detection and short-term tracking. However, by the time the action detection works in this

dissertaion was performed, combining the boosting and the particle filter is still a good choice because there are no regional proposal networks at that time.

1.3 Outline

In the following, the organization of this dissertation is

Chapter 2. Video Representation Related Works and Tools Used.

Chapter 3. Image-set Representation of a Face Video as a Set of Deep Features.

Chapter 4. Image-set Representation of Face Videos via Inter-frame Models.

Chapter 5. Temporal Representation of Egocentric Videos via Motion Models.

Chapter 6. Spatiotemporal Representation of Human Action Videos as 3D Nets.

Chapter 7. Supervised Hashing via Learning Embedding and Quantization.

Chapter 8. Conclusion.

Chapter 2

Video Representation Related Works and Tools Used

General speaking, this dissertation studies video-based action recognition and action quality assessment. It studies video representation in the computer vision perspective. As a result, the tasks are more about inferring the appearance, motion, geometry, and semantics than manipulating the data modality itself. However, still borrows tools from related communities such as signal processing and machine learning. While the problems are computer vision problems, the tools to solve the problem are not all well established in computer vision. For example, regarding 3D vision problems, it mainly refers to affine transformation, coordinate transformation, motion model, rotation angle, quaternion, camera model and essential matrix. These tools have been well established in computer vision. However, a counterexample is

compressed sensing and deep learning. During the development of the dissertation works, these fields are still evolving, particularly for deep learning.

2.1 Related works of video representation

While there is no temporal modeling at all, the discrete images representation may be sufficient if our interest is to know about individual frames.

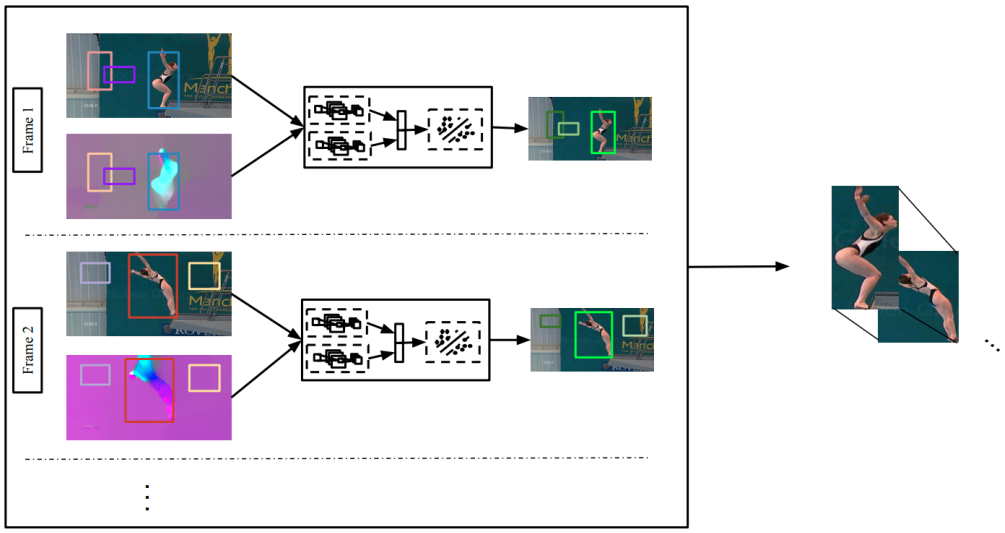
2.1.1 Image-set models

Set theory is a branch of mathematical logic that studies sets, which informally are collections of objects. Canonical representations of image sets include the covariance matrix [23] which is widely adopted in video recognition tasks such as object tracking [23, 24], facial expression recognition [25] and modeling dynamic texture [26]. Given frame-wise feature set $F = \{f_1, f_2, ..., f_n\}(f_i \in \mathbb{R}^d)$, the covariance matrix is $C = \frac{1}{n-1}\sum_{i=1}^{n}(f_i - \hat{f})(f_i - \hat{f})^T$ where $\hat{f}$ is the feature mean. When frames are shuffled, the covariance matrix of per-frame features does not change.

Notably, another representation is image set based collaborative representation [27] yet mostly for really a set of non-consecutive images such as the query and gallery sets for face recognition. Treating video as a set of frames is still an exploitative approach. For example, R. Wang *et. al.* represent all frames for each video clip as an image set, which can be modeled as a linear subspace to be embedded in

Grassmannian manifold [28]. Suppose $f_1, f_2, ..., f_n$ follow a k-dimensional Gaussian $\mathbf{N}(\mu, \Sigma)$ where $\mu = E(f_i) = \frac{1}{n}\sum_{i=1}^{n} f_i$ and $\Sigma = E[(f_i - \mu)(f_i - \mu)^T] = \frac{1}{n-1}(f_i - \hat{f})(f_i - \hat{f})^T$. As the basic idea, the similarity of two subspaces can be measured via mapping Grassmann manifold to Euclidean space by Projection kernel originated from principle angles. By representing image sets as linear subspaces lying on Grassmann manifold, recent studies have proposed to embed the Grassmann manifold into a high-dimensional kernel Hilbert space by exploiting the Project Metric. To overcome the limitations from kernel-based methods, *e.g.*, implicit map and high computational cost, Ruiping Wang *et. al.* propose a novel method to learn the Projection Metric directly on Grassmann manifold rather than in Hilbert space [29]. While they explored that mostly for face videos, A. Rahimi *et. al.* have attempted to learn appearance manifold that for simple human actions [30].

Image-set based video models can be roughly summarized into sample-based models [31, 32], subspace-based models [33, 34] and distribution-based models [35]. In sample based models, image sets are compared based on matching their sample-based statistics such as sample mean and convex combination of samples. In subspace-based models, subspace-based statistics are estimated to model image sets and then classify them with a given similarity function. In the distribution-based models such as the Gaussian mixture models [35], each image set is modeled with distribution-based statistics (*e.g.*, Gaussian distribution). Then the similarity between two distributions is measured in terms of the Kullback-Leibler Divergence.



(a) Action detection at every frame of the video (b) Linked detections in time produce *action tubes*

Figure 2.1: This figure is from [3], shown for the 10m platform diving. (a) Candidate regions are fed into action specific classifiers, which make predictions using static and motion cues. (b) The regions are linked across frames based on the action predictions and their spatial overlap. Action tubes are produced for each action and each video.

For action localization and segmentation, image-based detection and segmentation methods can be adapted to image set in a straightforward way - frame by frame object/action detection and segmentation. However, the temporal coherence may not be preserved. These video tasks can also be attacked using spatiotemporal models.

2.1.2 Locally temporally motion models

Long-term robust visual tracking is still a challenge, primarily due to the appearance changes of the scene and target. In the following, the recent progress is briefly reviewed regarding image representation, appearance model and motion model for

building a tracking system. For details, please also refer to [36] and references therein. This review has covered most tracking methods until the rise of deep learning.

Appearance models. For examples, (1) Histogram or Parzen estimator model. Their merit is that when photometry changes, the histograms bin or Parzen estimators kernel changes with photometry variation. Also, they are resilient to blur and resolution variations. But they are invariant to deformations and arbitrary permutation of pixels. The histogram does not limit to the color histogram. The edge, orientation, texture and motion histograms can also be used. (2) Additionally, template matching [37]. The basic idea is to use a cropped image of the target from the initial frame to represent the appearance of the target. It is based on the assumption that the object will look nearly identical in each new image and that movement is a nearly pure 2D translation. As regards changes in the photometry, Normalized Cross Correlation (NCC) can also be used to add some resilience. However, it is easy to encounter the drift problem [38]: if the estimate of template location is slightly off, then it searches for a matching position that is similarly off the center. Over time, this offset error builds up until the template starts to slide off the object. In [38], Matthews *et al.* have proposed a strategy for template update with drift correction.

Motion models. For examples, (1) The optical flow method is based on the assumption of constant lightness across frames. That is true if the illumination condition does not have drastic change or the frame rate is high. The Lucas-Kanade method [39] is essentially a template tracker that works by gradient descent and can be generalized

to other 2D parametric motion models (*e.g.*, affine, projective one) [40]. Its implementation by tracking Shi-Tomasi corner features [39] is re-named Kanade-Lucas-Tomasi method (KLT), a widely-used motion estimation method. However, this assumption for all pixels in a large template is unreasonable. (2) the Bayesian filtering framework can be used to model motion. In the Bayesian filtering framework [41] (*e.g.*, Kalman filter, particle filter), it is wanted to recursively estimate the current target state vector each time a new observation is received.

2.1.3 Globally temporal models

Globally, there exist recurrent neural networks and particularly the Long Short-Term Memory network off-the-shelf. Splitting videos into multiple short clips is likely to destroy the long-term pattern. The Temporal Convolutional Networks [42] (TCN) perform 1-D convolution over the per-frame feature vectors along the temporal dimension. A simplified version without recurrence is building additional 1D CNN on top of the second-last layer of 2D CNNs across frames such as the Temporal Convolutional Network (TCN) [43]. TCN [42, 43] has achieved the state-of-the-art performance on segmenting fine-grained actions. The input to TCN is a sequence of 1D features normally extracted by 2D CNN such as the ResNet [44] or DenseNet [45] for every frame. Encoder-Decoder TCN (ED-TCN) [42] does a downsampling (encode) and upsampling (decode), and after the softmax at the top layer, will predict the action class for that frame. It works well for action segmentation which is more or less decided

by the difference across frames in a single video.

However, for video classification across various videos, a video descriptor is also needed to characterize the video in a comprehensive and compact way. TCN operates in the feature domain, which means it captures the correlation of feature vectors. A video definitely contains temporal information such as motion. It could hardly say where the motion cue is captured once the input to TCN is feature vectors and the feature vectors are appearance based. A feature vector is nothing but a list of numbers. As the numbers are normally the output of a softmax layer in CNNs, they more or less means the likelihood of an image belonging to a certain category indicated by the index of that number. However, the label categories of 2D CNNs are normally object or scene categories rather than video or action categories. The label consistency is important. If our prediction is video or action category, it would be ideal if the training data are also labeled with video or action categories, even for the prepossessing step of input feature generation. It would be arbitrary to let the video or action category be decided by the numbers of how likely a frame is a cat or the sky. Instead, it is more convincing if the features are the output of spatiotemporal CNNs that are trained with the labels of video or action categories. For action segmentation, as long as the feature vector characterize each frame well, there is no big impact if the feature vector does not contain motion information. It is because the correlation in the feature domain represents the correlation in the spatial domain since there is a good appearance-based feature descriptor. However, for action recognition, it is

definitely desired that the feature descriptor to capture the temporal correlation in the spatial domain in terms of the raw pixel intensities which is how optical flow is calculated. It is hard to understand how the correlation in the feature domain can capture the optical flow.

Temporal networks are normally unable to capture motion cues until Two-Stream that additionally trains the same network on pre-computed optical flow images. Two-Stream Convolutional Networks [46] (Two-Stream CNNs) joint trains two similar networks with inputs of raw frames and optical flows, respectively. The spatial part, in the form of individual frame appearance, carries information about scenes and objects depicted in the video. The temporal part, in the form of motion across the frames, conveys the movement of the observer (the camera) and the objects. The fusion of the two nets is a late fusion with pre-trained models. Two-Stream has become an effective network to encode both the appearance and motion information of videos. For example, [47] proposes a Two-Stream RNN architecture to model both temporal dynamics and spatial configurations for skeleton (joint) based action recognition. They explore two different structures for the temporal stream: Stacked RNN and Hierarchical RNN. Hierarchical RNN is designed according to human body kinematics. They also propose two effective methods to model the spatial structure by converting the spatial graph into a sequence of joints. The fusion is implemented by combining the softmax class posteriors from the two networks. [48] combines the Two-Stream Networks with learnable spatiotemporal feature aggregation. The resulting architecture is end-to-

end trainable for whole-video classification. [49] develops a CNN architecture that is trainable in an end-to-end manner directly for the place recognition task. The main component of this architecture, NetVLAD, offers a powerful pooling mechanism with learnable parameters that can be easily plugged into any other CNN architecture. Furthermore, NetVLAD is a new generalized VLAD layer, inspired by the Vector of Locally Aggregated Descriptors (VLAD) [50] image representation commonly used in image retrieval.

Among the temporal deep networks inputted with deep spatial features, other than TCN, [51] embeds the Deep Temporal Linear Encoding Networks inside ConvNet architectures, aiming to aggregate information from an entire video, be it in form of frames or clips. The result is a global feature representation obtained in an end-to-end learning scheme. [52] integrates ideas from attention models and gated recurrent networks to better deal with noisy or unsegmented sequences. The attention modeling and recurrent hidden representation learning are separated as two independent models. [53] select keyframes and then compute deep features for them in a joint manner with an optical flow field.

2.1.4 Locally spatiotemporal models

In terms of spatial localization - the so-called action tube or tubelet proposal, there exist a number of works [3, 19, 54] (see Fig. 2.1). In [19], class-independent tubelet proposals are fisted generated and then inputted into a class-specific (spatio-

) temporal network such as C3D and LSTM. There are many association methods proposed for linking 2D regional proposals to form a 3D tubelet proposal [19, 54]. The cross-frame regional proposal linking problem is essentially same with the data association problem in object tracking algorithms [55] and similar with the object-person linking problem for different proposals within the same image [56].

3D CNN is not particularly designed for video analytics. The patch-level 3D CNN has become a hardcore for medical image analysis[1]. In detail, 3D CNNs have been adapted to process 3D volumetric medical images in a straightforward way - 3D kernels for convolutional layers [57, 58]. The 3D conversion is similar for pooling layers and fully-connected layers.

However, 3D CNN is just a surrogate of the spatiotemporal network that is highly desired in the video analytics community [59–61] since there is an asymmetry between space and time. Time is intrinsically different from space[2].

If a 3D CNN is trained for a preset number of frames, then it expects that number of frames at testing. The case is also true for fine-tuning pre-trained networks. For example, similar with the pre-trained C3D CNN [61] used in the previous work [62], the pre-trained P3D CNN that is adapted can only have an input size of 16 frames as well. There are two common reasons for setting a relatively tight temporal length: locality and memory. First, compared with 2D CNN capturing spatial local connectivity, 3D CNNs for videos are intuitively designed to capture temporally local motion

[1] `https://luna16.grand-challenge.org/results/`
[2] `https://en.wikipedia.org/wiki/Arrow_of_time`

and coherence. Second, it is a practical challenge for the GPU memory to store all the weights of a long-term 3D CNN considering scaling up all the 2D CNN's parameters by the number of frames. For example, the model size of an 11-layer C3D CNN is even larger than that of a 235-layer Residual Network (ResNet) [44]. The Inflated 3D (I3D) [63] has been released by the Kinetics creator and trained with 64-frame inputting clips. But it is unable to fine-tune it due to lack of memory.

Among the spatiotemporal deep networks, in [64] Sigurdsson *et. al.* proposes a deep-structured model using a fully-connected temporal CRF that not only models semantic aspects of activities but also reasons about long-term temporal relations. It is verified in the tasks of video classification and temporal localization. In [65], Feichtenhofer *et. al.* proposes a general spatiotemporal ConvNet, T-ResNet, based on transforming a purely spatial network to one that can encompass space-time via hierarchical injection of temporal residuals. It works like a revised version of ResNet. They also apply the temporal residual units to the appearance and flow networks of a two-stream architecture. In [66], Jain and Zamir *et. al.* proposes Structural-RNN (S-RNN) for doing deep learning over spatio-temporal graphs. In [67], Garcia-Hernando *et. al.* introduces a novel method called transitions forests, an ensemble of decision trees that both learn to discriminate static poses and transitions between pairs of two independent frames. The proposed training procedure helps to capture temporal dynamics in a more effective way than other strong forest baselines. Aggregation models such as Soft Bag-of-words, Fisher Vectors, NetVLAD, GRU, and LSTM have

been examined in the literature [68]. They introduced the Context Gating mechanism and have shown its benefit for the trainable versions of BoW, VLAD, and FV. The ensemble of the individual models has been shown to improve the performance further, enabling their method to win the Youtube 8M Large-Scale Video Understanding Kaggle challenge. In [48], the proposed new layer ActionVLAD is a spatiotemporal extension of the NetVLAD aggregation layer [49] that has been shown to work well for instance-level recognition tasks in still images.

2.2 Tools used in this dissertation

In the recent decade when I step into the field of visual computing (*i.e.*, visual information processing), there has been great progress pushed by a branch of statistical signal processing (*a.k.a.*, compressed sensing and sparse recovery) and a branch of machine learning (*a.k.a.*, deep neural networks). Models get more and more advanced and understandings about these models get deeper and deeper. In the former few years of my dissertation works, I follow the compressed sensing research closely. In the latter few years, namely the recent years, tools in deep learning are also studied to address video recognition problems.

2.2.1 Compressed sensing and sparse recovery

Compressed sensing is an area in signal processing. One goal of compressed sensing and sparse recovery is to answer how to find underlying structures in noisy observations. For example, originally it is posed to help reconstruct sharp and high-quality MR images. In the recent decade, the idea of sparsity has driven fruitful results in regression analysis, harmonic analysis and compressed sensing [69–79]. For details, please see also the course webpage [3] of JHU's EN.520.648 Sparse Recovery and Compressed Sensing.

When observing a random signal $\mathbf{y}$ for recognition, it is hoped to send the classifier a *discriminative compact* representation $\mathbf{x}$, which satisfies $\mathbf{A}\mathbf{x} = \mathbf{y}$ and is yet computed by pursuing the best *reconstruction*. It could be coding a batch of observations of the observed variable $\mathbf{y}$ itself, *i.e.*, $\mathbf{A} = \mathbf{Y} = \left[\mathbf{y}^{(1)}|...|\mathbf{y}^{(\tau)}\right]$, such as subspace clustering. When $\mathbf{A}$ is under-complete, $\mathbf{x}$ is a dimension-reduced representation of $\mathbf{Y}$. But all data in $\mathbf{A}$ are useful and the linear system is over-determined. Although the exact solution is unique if existing, it may even not exist. However, a closed-form approximate solution can be obtained by minimizing the reconstruction error and namely Least-Squares:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 \approx (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y}.$$

When $\mathbf{A}$ is over-complete, There exists a solution that is adding a Tikhonov regularizer [80]:

[3] `http://thanglong.ece.jhu.edu/Course/648/`

$\mathbf{x}^* = \arg\min_{\mathbf{x}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda_r \|\mathbf{x}\|_2^2 = \arg\min_{\mathbf{x}} \|\widetilde{\mathbf{y}} - \widetilde{\mathbf{A}}\mathbf{x}\|_2^2$

$\approx (\mathbf{A}^T\mathbf{A} + \lambda_r\mathbf{I})^{-1}\mathbf{A}^T\mathbf{y}$ where $\widetilde{\mathbf{y}} = [\mathbf{y}, \mathbf{0}]^T$; $\widetilde{\mathbf{A}} = [\mathbf{A}, \sqrt{\lambda_r}\mathbf{I}]^T$ is always under-complete. But $\mathbf{x}^*$ is not necessarily compact yet generally dense. $\mathbf{x}$ is a dimension-augmented representation of $\mathbf{y}$. If it is insisted on applying Least Squares, the augmented dimension will be hallucinated. Nontheless, since the data in $\mathbf{A}$ are redundant, Alternatively, a sparse representation of $\mathbf{A}$ can be pursued. In a supervised way, Sparse Representation based Classification [73] (SRC) expresses a test sample $\mathbf{y}$ as a linear combination $\mathbf{y} = \mathbf{D}\mathbf{x}$ of all training samples in a dictionary $\mathbf{D}$. *Since non-zero coefficients should all drop to the ground-truth class, ideally not only* $\mathbf{x}$ *is sparse but also the class-level sparsity is* 1. When enforcing an **atom-wise** sparsity, it is also hoped that the ground-truth class dominates the supports (*i.e.*, **class-wise** sparsity). In fact, non-zero coefficients also drop to other classes due to noises and correlations among classes. By adding a sparse error term $\mathbf{e}$, SRC simply employs an atom-wise sparsity:

$$[\mathbf{x}^*, \mathbf{e}^*]^T = \arg\min_{\widetilde{\mathbf{x}}} sparsity(\widetilde{\mathbf{x}})$$
$$s.t. \qquad \mathbf{y} = \mathbf{D}\mathbf{x} + \mathbf{e} = [\,\mathbf{D} \mid \mathbf{I}\,] \times \begin{bmatrix} \mathbf{x} \\ \\ \mathbf{e} \end{bmatrix} = \widetilde{\mathbf{D}}\widetilde{\mathbf{x}},$$

where $\widetilde{\mathbf{D}}$ is over-complete and thus needs to be sparsely used. Once a compact $\mathbf{x}$ is available for classification, SRC evaluates which class leads to the minimum reconstruction error, which can be seen as a max-margin classifier [81]. Using a fixed $\mathbf{D}$ without dictionary learning [82] or sparse coding, SRC still performs robustly well for

denoising and coding tasks such as well-aligned noisy face identifications. But there is a lack of theoretical justification why a sparser representation is more discriminative. [72] incorporates the Fisher's discrimination power into the objective. [83] follows the regularized Least-Squares [80] and claims SRC's success is due to the linear combination as long as the ground-truth class dominates coefficient magnitudes. SRC's authors clarify this confusion using more tests on robustness to noises [84].

In practice, it cares more about how to recover $\mathbf{x}$ [85]. Enforcing sparsity is feasible since $\mathbf{x}$ can be exactly recovered from $\mathbf{y} = \mathbf{D}\mathbf{x} + \mathbf{e}$ under conditions for $\mathbf{D}$ [71]. However, finding the sparsest solution is NP-hard and difficult to solve exactly [86]. But now, it is well-known that the ℓ_1 norm is a good convex relaxation of sparsity -- minimizing the ℓ_1 norm induces the sparsest solution under mild conditions [70]. Exact recovery is also guaranteed by ℓ_1-minimization under suitable conditions [69]. Typically, an iterative greedy algorithm is the Orthogonal Matching Pursuit (OMP) [85]. A number of efficient ℓ_1-minimization algorithms have been adapted from the convex optimization literature such as Augmented Lagrangian Method (ALM) [87].

For multichannel $\mathbf{Y}$ with dependant coefficients across channels over time [88], $\mathbf{Y} = \mathbf{D}\mathbf{X}$ where $\mathbf{X}$ is low-rank. Principal Component Analysis of $\mathbf{Y}$ is $\min_{\mathbf{A}} \|\mathbf{Y} - \mathbf{A}\mathbf{Y}\|^2$ where $\mathbf{A}$ is a projection matrix and Sparse Subspace Clustering [79] involves solving $\mathbf{Y} = \mathbf{Y}\mathbf{X}$ where $\mathbf{X}$ is sparse.

Matrix approximation. Matrices with low numerical rank appear in principal componnet analysis (PCA), which is exactly low-rank matrix approximation, as shown

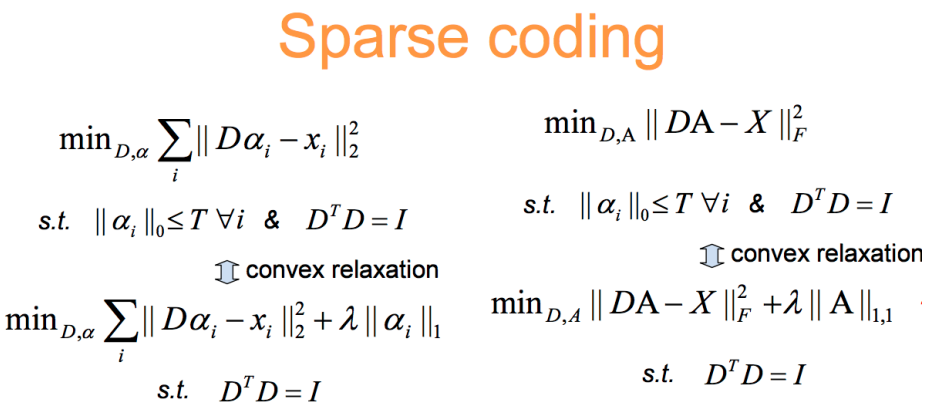


Figure 2.2: Formulation of sparse coding using the same notation with PCA's.

in Fig. 2.3.

Sparse coding has its root in neuroscience and has been well exploited in harmonic analysis, signal processing, and compressed sensing.

Convex optimization. Matching Pursuit dates back to 1993 in [89] and Recursive Matching Pursuit for sparsely coding Multi-Measurement Vectors can be traced back to 1998 in [90].

Structured sparsity regularization generalizes and extends sparsity regularization in sparse coding, by allowing for optimal selection over structures like groups.

2.2.2 Representation learning

Representation learning is an area in machine learning. A long-standing problem in statistics and related areas is how to find a suitable representation of multivariate data. Representation here means that somehow the data is transformed so that its

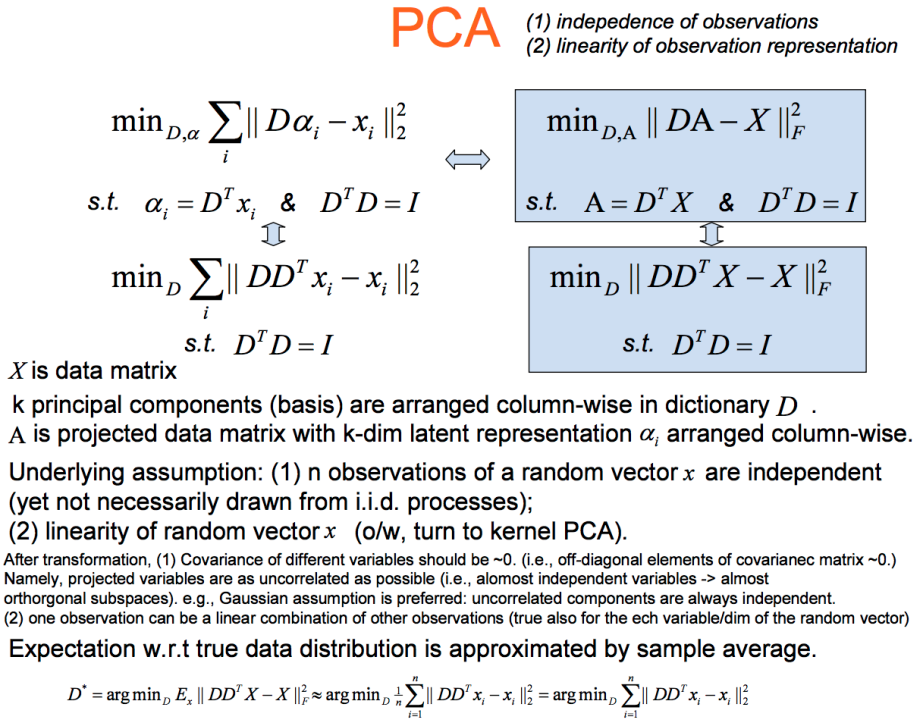


Figure 2.3: Illustration of Principal Component Analysis.

essential structure is made more visible or accessible. In terms of representations of faces, the most principal component is called Eigenface [91] - the face representation obtained from Eigen decomposition. For details, please see also the course webpage [4] of JHU's EN.600.479/679 Representation Learning.

Principal Component Analysis (PCA). PCA is one of the most used representation learning models and also used for dimensionality reduction. It is well known that the principal components of a random vector are the top-k eigenvectors of the covariance (auto-correlation) matrix. This is the statistical illustration of PCA. However, the intuitive way to explain PCA is from the geometric perspective. It purses a pro-

[4] `http://www.cs.jhu.edu/~raman/Courses/CS679f14.html`

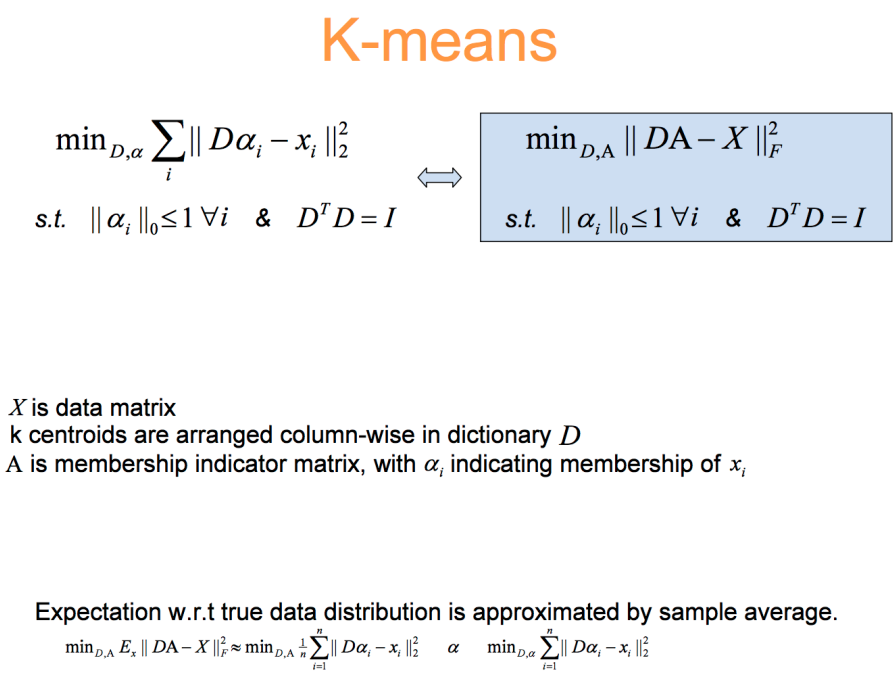


Figure 2.4: Formulation of K-means clustering.

jection to project the high-dimensional data into the low-dimensional space where the total reconstruction error of all data points is the smallest. From the view, there also exists a rank minimization perspective. Projected data points in the low-dimensional space form the low-rank subspace compared to original high-dimensional space.

Sparse representation is interchangeable with our **Sparse coding** in the context of this dissertation. This has been explained previously in Fig. 2.2.

K-means clustering. K-means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem.

Linear discriminant analysis (LDA). LDA aims to find a discriminant function that is either linear in the components of the original data, or linear in some given

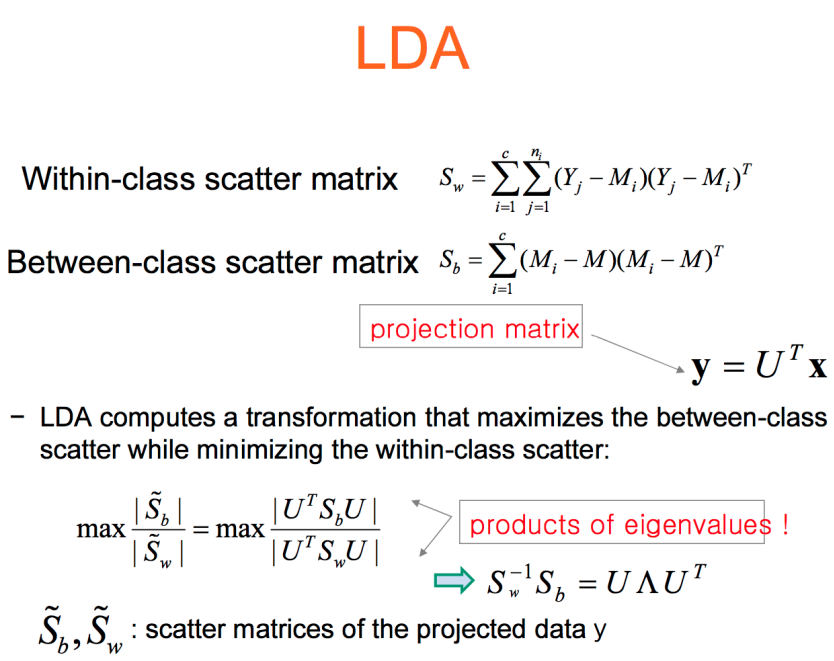


Figure 2.5: Formulation of Linear Discrimination Analysis.

set of functions of the data. The problem of finding a linear discriminant function can be formulated as a problem of minimizing a criterion function, such as using the so-called the between-class distances and within-class distances used in Fisher's linear discriminant. In terms of representations of faces, the LDA version is called Fisher face [91], in corresponding to Eigenface in the PCA setting.

Independent component analysis (ICA). The problem of blind source separation boils down to finding a linear representation in which the components are statistically independent. Independent component analysis (ICA) is a method for finding underlying factors or components from multivariate (multidimensional) statistical data. The projection pursuit and sparse coding connections are related to a deep result that

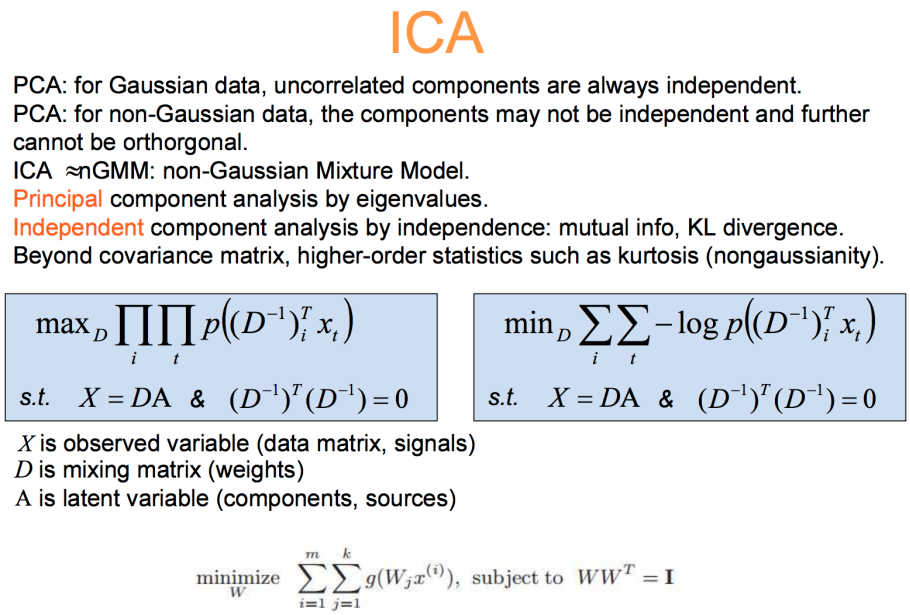


Figure 2.6: Formulation of the Independent Component Analysis.

says that ICA gives a linear representation that is as structured as possible.

Neural Network. A typical example of fully-connected Neural Networks is perceptron, without convolution as in Convolutional Neural Network or missing connections as in the modern Dropout and DropConnect. In perceptron as shown in Fig. 2.8, the input, hidden, and output variables are represented by nodes, and the weight parameters are represented by links between the nodes. The arrow of the links indicates the direction of information flow through the network during forward propagation.

Convolutional Neural Networks (CNN). Unlike a regular Neural Network, convoluational layers are introduced into CNN. CONV layer computes the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input

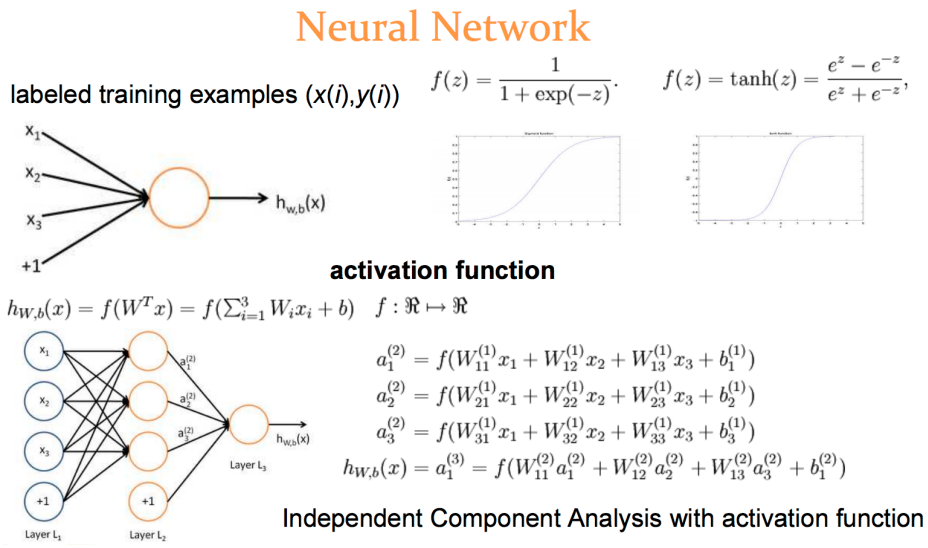


Figure 2.7: Illustration of the neural network using simple networks.

volume. Together with CONV layers, pool layers are also introduced, as shown in Fig. 2.8.

Deep CNN is the type of CNN with many layers, namely with a large number of parameters. To accomplished this, Krizhevsky *et. al.* proposed to use a rectified linear units (ReLUs) activation function. The architecture, also known as AlexNet, consists of 5 CONV layers and 3 fully connected layers.

Recurrent Neural Network (RNN). RNN is deep in time and thus can be treated as a deep neural network with the issue of vanishing gradients. Long Short-Term Memory network [92] dating back to 1997 is a type of RNN that gets around of vanishing gradient problem. Note that there also exists a type of network called Recursive Neural Network [93] which is generalized RNN with a deep structure of a skewed tree. There are connections among all those models and hidden Markov

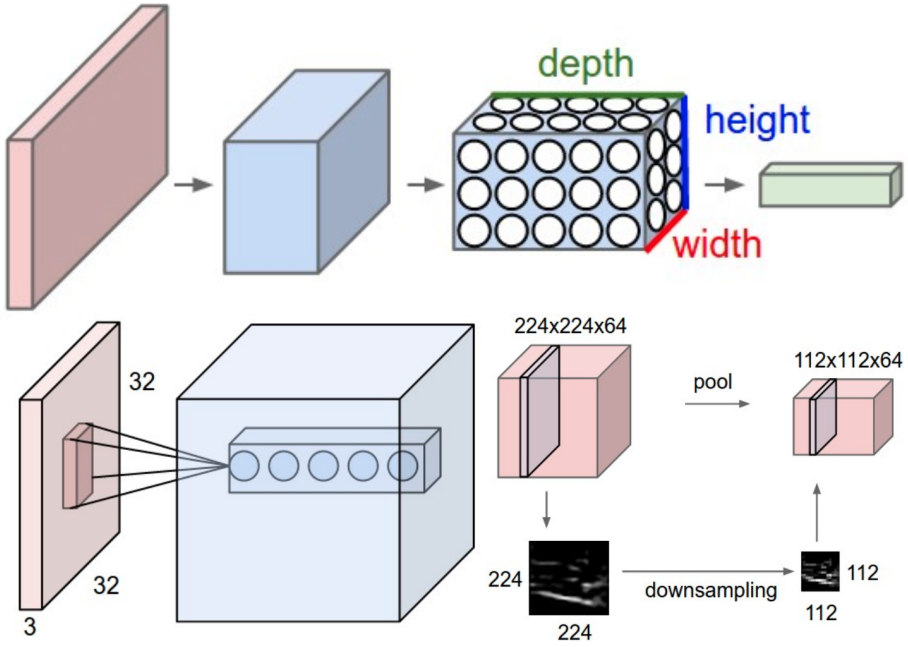


Figure 2.8: Illustration of the convolutional layer and pooling layer in CNN. Figures composed from pictures on the Internet.

models. Both a feed-forward multilayer neural network and a hidden Markov model can be seen as a directed acyclic graph with hidden nodes. Both a recurrent neural network and a hidden Markov model map a sequence of inputs to a sequence of outputs via a sequence of hidden states. Long Short-Term Memory learns a function of inputs and hidden states using a perceptron-like network with gating weights further learned using perceptrons.

Factor analysis. In this dissertation, it means studying each source of variation separately and keeping all other variations as constants in a control experiment. It is related with PCA.

Chapter 3

Image-set Representation of a Face Video as a Set of Deep Features

From the previous section, it can be seen in the literature that a video can be represented as a set of images. Now in the deep learning era, the representation can further be a set of deep features which are generated using deep CNNs for each frame. For example, Fig. 3.1 is a not a problem of the time series prediction although the curve is plotted in the order of frames. Instead it is a regression problem - fitting the known $(frame, intensity)$ in training and then use the fitted curve to infer a new $(frame, intensity)$ in testing. By treating a video as an image set, set-based methods have made great progress on video-based facial recognition [94].

This chapter first deals with two challenges for measuring the similarity of the subject identities in practical video-based face recognition - the variation of the head pose

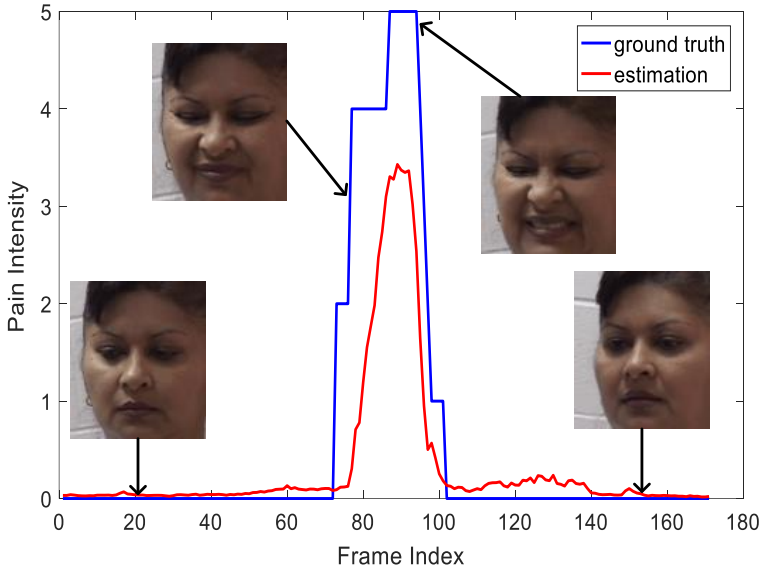


Figure 3.1: Representing a facial expression video as a set of images, each of which associates with an expression intensity. The continuous red curve displays the estimated pain intensities. The blue curve is connected from discrete points of $(frame, intensity)$ where the $intensity$ is per-frame observer-rated label.

in uncontrolled environments and the computational expense of processing videos. Since the frame-wise feature mean is unable to characterize the pose diversity among frames, the model preserves the overall pose diversity and closeness in a video. Then, identity will be the only source of variation across videos since the pose varies even within a single video. Instead of simply using all the frames, the model selects those faces whose pose point is closest to the centroid of the K-means cluster containing that pose point. Then, the model represents a video as a bag of frame-wise deep face features while the number of features has been reduced from hundreds to K. Since the video representation can well represent the identity, now it measures the subject similarity between two videos as the max correlation among all possible pairs in the two bags of features. On the official 5,000 video-pairs of the YouTube Face dataset for

face verification, our algorithm achieves a comparable performance with VGG-face that averages over deep features of all frames. Other vision tasks can also benefit from the generic idea of employing geometric cues such as 3-D poses to improve the descriptiveness of deep features learned from appearances.

However, in order to train deep networks, limited labeled data are available for the research of estimating facial expression intensities. For instance, the ability to train deep networks for automated pain assessment is limited by small datasets with labels of patient-reported pain intensities. Fortunately, fine-tuning from a data-extensive pre-trained domain, such as face verification, can alleviate this problem. In the following, the second section of this chapter proposes a network that fine-tunes a state-of-the-art face verification network using a regularized regression loss and additional data with expression labels. In this way, the expression intensity regression task can benefit from the rich feature representations trained on a huge amount of data for face verification. The proposed regularized deep regressor is applied to estimate the pain expression intensity and verified on the widely-used UNBC-McMaster ShoulderPain dataset, achieving the state-of-the-art performance. A weighted evaluation metric is also proposed to address the imbalance issue of different pain intensities.

3.1 Key subset selection with collaborative aggregation for face videos

In this section, the generic task is measuring the similarity of one source of variation among videos such as the subject identity in particular. The motivation of this work is as followed. Given a face video visually affected by confounding factors such as the identity and the head pose, it can be compared against another video by hopefully only measuring the similarity of the subject identity, even if the frame-level feature characterizes mixed information. Indeed, deep features from Convolutional Neural Networks (CNN) trained on face images with identity labels are generally not robust to the variation of the **head pose**, which refers to the face's relative orientation with respect to the camera and is the primary challenge in uncontrolled environments. Therefore, the emphasis of this section is not the deep learning of frame-level features. Instead, it is of interest regarding how to improve the video-level representation's descriptiveness which rules out confusing factors (*e.g.*, pose) and induces the similarity of the factor of interest (*e.g.*, identity).

If the model treats the frame-level feature vector of a video as a random vector, it may be assumed that the highly-correlated feature vectors are identically distributed. When the task is to represent the whole image sequence instead of modeling the temporal dynamics such as the state transition, the sample mean and variance may be used to approximate the true distribution, which is implicitly assumed to be a

normal distribution. While this assumption might hold given natural image statistics, it can be untrue for a particular video. Even if the features are Gaussian random vectors, taking the mean makes sense only if the frame-level feature just characterizes the identity. Because there is no variation of the identity in a video by construction. Hover, even the CNN face features still normally contain both the identity and the pose cues. Surely, the feature mean will still characterize both the identity and the pose. What is even worse, there is no way to decouple the two cues once the mean is taken. Instead, if the video feature is wanted to only represent the subject identity, the model had better preserve the overall pose diversity that very likely exists among frames. Disregarding minor factors, the identity will be the only source of variation across videos since pose varies even within a single video. Then, following such a disentangling variation idea, a keyframe selection algorithm is proposed to retain those keyframes that preserve the pose diversity. Based on the selection, an algorithm is further designed to compute the identity similarity between two sets of deep face features by pooling the max correlation.

Fig. 3.2 shows an example video snippet lasting only a couple of seconds in the YouTube Face (YTF) dataset [95]. A 1-minute video easily gets to two thousand frames. Why bother to send all of them to CNN when they look so similar? Instead of pooling from all the frames, the proposed K frame selection algorithm is highlighted at firstly the pose quantization via K-means and then the pose selection using the pose distances to the K-means centroids. It reduces the number of features from

Figure 3.2: Example of the chosen key faces. Top row shows the first 10 frames of a 49-frame YTF sequence of Woody Allen, who looks right and down sometimes. And most of the time his face is slightly slanting. Bottom row is 9 frames selected according to the variation of 3D poses. Disclaimer: the source owning this YouTube video allows republishing the face images.

tens or hundreds to K while still preserving the overall pose diversity, which makes it possible to process a video stream at real time. Thus, this algorithm also samples the video frames (to K images). Once the keyframes are chosen, the network will pool a single number of the similarity between two videos from many pairs of images. The metric to pool from many correlations normally is the mean or the max. Taking the max is essentially finding the nearest neighbor, which is a typical metric for measuring similarity or closeness of two point sets. In our work, the max correlation between two bags of frame-wise CNN features is employed to measure how likely two videos represent the same person. In the end, a video is represented by a single frame's feature which induces nearest neighbors between two sets of selected frames if each frame is treated as a data point. This is essentially a pairwise max pooling process. On the official 5000 video-pairs of YTF dataset [95], our algorithm achieves

a comparable performance with state-of-the-art that averages over deep features of all frames.

In short, the section introduces a pose-based frame selection method (Sec. 3.1.2) with a similarity max pooling for the problem of face recognition in videos (Sec. 3.1.3). Particularly, this section first uses the head pose estimation to analyze the faces within videos, and then extends K-means algorithm to select K keyframes which has representative poses in the videos. In the end, a max pooling is employed to ensemble the similarities of selected keyframes. In Sec. 6.2.6, the experiment on YTF verifies the proposed method behaves comparably with VGG-face that is developed by the Oxford VGG.

3.1.1 Similarity measure related works

Measuring the similarity of subject identity is useful face recognition such as face verification and face identification. Face verification is to decide whether two modalities containing faces represent the same person or two different people. Thus, it is important for access control or re-identification tasks. Face identification involves one-to-many similarity, namely a ranked list of one-to-one similarity and thus is important for watch-list surveillance or forensic search tasks. In identification, information is gathered about a specific set of individuals to be recognized (*i.e.*, the gallery). At test time, a new image or group of images is presented (*i.e.*, the probe). Essentially, measuring if two videos represent the same person is comparing the similarity of two

image sets. One measure is how overlapping the elements of each set are.

In the deep learning era, face verification on a number of benchmarks such as the Labeled Face in the Wild (LFW) dataset [96] has been well solved by DeepFace [97], DeepID [98], FaceNet [4] and so on. The Visual Geometry Group at the University of Oxford released their deep face model called VGG-Face Descriptor [99] which also gives a comparable performance on LFW. However, in the real world, pictures are often taken in an uncontrolled environment (the so-called in the wild versus in the lab setting). Considering the number of image parameters that were allowed to vary simultaneously, it is logical to consider a divide-and-conquer approach - studying each source of variation separately and keeping all other variations as constants in a control experiment. Such a separation of variables has been widely used in Physics and Biology for multivariate problems. In this data-driven machine learning era, it seems fine to remain all variations in realistic data, given the idea of letting the deep neural networks learn the variations existing in the enormous amount of data. For example, FaceNet [4] trained using a **private** dataset of over 200M subjects is indeed robust to poses, as illustrated in Fig. 3.3. However, the CNN features from conventional networks such as DeepFace [97] and VGG-Face [99] are normally not. Moreover, the unconstrained data with fused variations may contain biases towards factors other than identity, since the feature might characterize a mixed information of identity and low-level factors such as pose, illumination, expression, motion, and background. For instance, pose similarities normally outweigh subject identity similarities, leading

to matching based on pose rather than identity. As a result, it is critical to decouple pose and identity. If the facial expression confuses the identity as well, it is also necessary to decouple them too. In the section, the face expression is not considered as it is minor compared with pose. Similarly, if it is wanted to measure the similarity of the face expression, it is also needed to decouple it from the identity. For example in [100] for facial expression recognition, one class of training data are formed by face videos with the same expression yet across different people.

Moreover, there are many different application scenarios for face verifications. For Web-based applications, verification is conducted by comparing images to images. The images may be of the same person but were taken at different time or under different conditions. Other than the identity, high-level factors such as the age, gender, ethnicity and so on are not considered in this section as they remain the same in a video. For online face verification, alive video rather than still images are used. More specifically, the existing video-based verification solutions assume that gallery face images are taken under controlled conditions [101]. However, the gallery is often built uncontrolled. In practice, a camera could take a picture as well as capture a video. When there is more information describing identities in a video than an image, using a fully live video stream will require expensive computational resources. Normally, video sampling or a temporal sliding window are needed.

3.1.2 Pose selection by diversity-preserving K-Means

This section explains our treatment particularly for real-world images with various head poses such as images in YTF. Many existing methods such as [100] make a certain assumption which holds only when faces are properly aligned.

By construction (say, face tracking by detection), each video contains a single subject. Each video is formalised as a set $\mathbb{V} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m\}$ of frames where each frame $\mathbf{x}_i$ contains a face. Given the homography $\mathbf{H}$ and correspondence of facial landmarks, it is entirely possible to estimate the 3D rotation angles (yaw, pitch and roll) for each 2D face frame. Concretely, some head pose estimator $p(\mathbb{V})$ gives a set $\mathbb{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\}$ where p_i is a 3D rotation-angle vector $(\alpha_{yaw}, \alpha_{pitch}, \alpha_{roll})$.

After pose estimation, the next step is to select keyframes with significant head poses. Our intuition is to preserve pose diversity while downsampling the video in the time domain. It can be learned from Fig. 3.3 of Google's FaceNet that face features learned from a deep CNN trained on identity-labeled data can be invariant to head poses as long as the training inputs for a particular identity class include almost all possible poses. That is also true for other minor sources of variations such as illumination, expression, motion, background among others. Then, identity will be the only source of variation across classes since any factor other than identity varies even within a single class.

Without such huge training data as Google has, instead it is hoped that the testing inputs for a particular identity class include poses as diverse as possible.

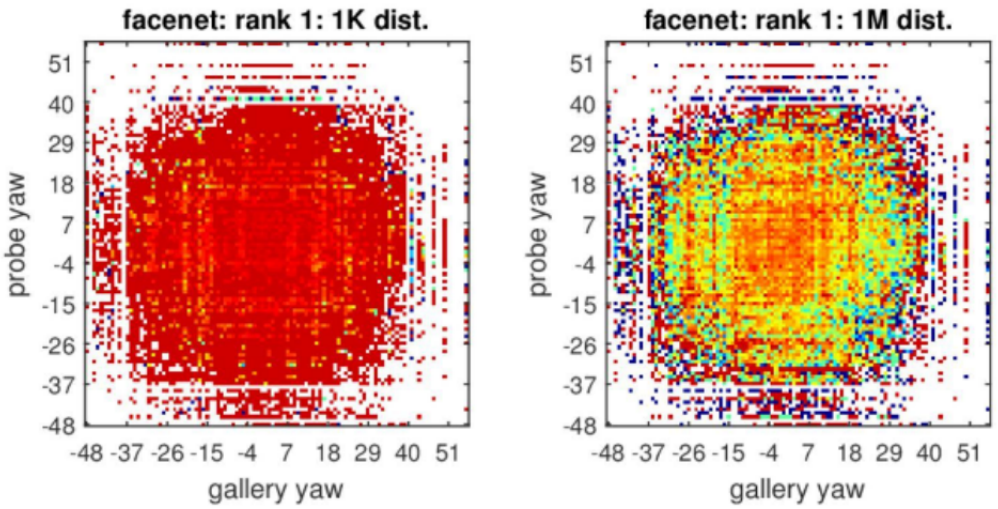


Figure 3.3: Analysis of rank-1 identification under varying poses for Google's FaceNet [4] on the recently established MegaFace 1 million face benchmark [5]. Yaw is examined as it is the primary variation such as looking left/right inducing a profile. The colors represent identification accuracy going from 0 (blue, none of the true pairs were matched) to 1 (red, all possible combinations of probe and gallery were matched). White color indicates combinations of poses that did not exist in the test set. (a) 1K distractors (people in the gallery yet not in the probe). (b) 1M distractors. This figure is adapted from MegaFace's FaceScrub results.

A straightforward way is to use the full video, which indeed preserves all possible pose variations in that video while computing deep features for all the frames is computationally expensive. Taking representing a line in a 2D coordinate system as an example, the model only needs either two parameters such as the intercept and gradient or any two points in that line. Similarly, now our problem becomes to find a compact pose representation of a testing video which involves the following two criteria.

First, the pose representation is compact in terms of non-redundancy and closeness. For non-redundancy, it is hoped to retain as few frames as possible but this

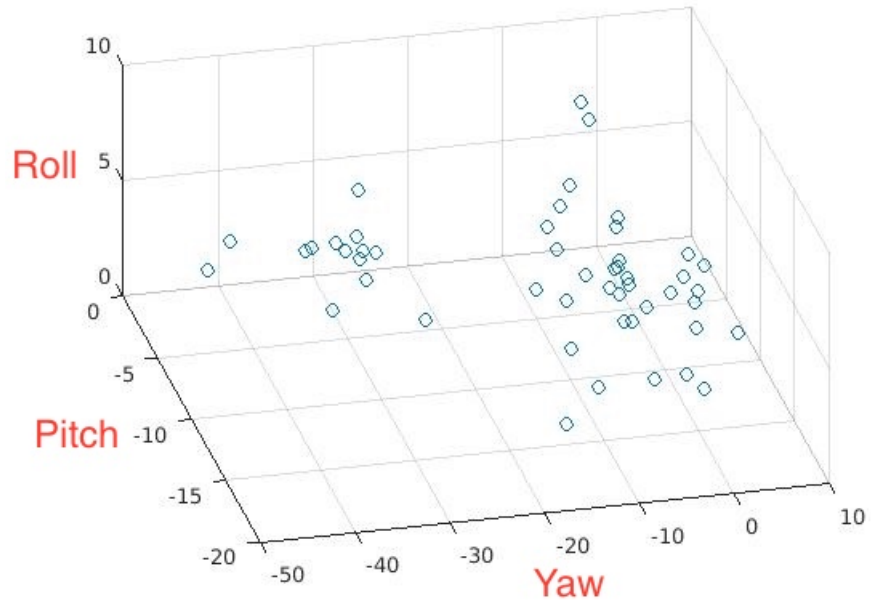


Figure 3.4: An example of 3-D pose space. Shown for the 49-frame Woody Allen sequence in YTF. Three axes represent rotation angles of yaw (looking left or right), pitch (looking up or down) and roll (twisting left or right so that the face is slanting), respectively. The primary variation is the yaw such as turning left/right inducing a profile. The pattern exists in pose distribution - obviously two clusters for this sequence so in extreme case for reducing computation it can set that $K = 2$.

criterion is not super critical. For pose closeness, it can be observed from Fig. 3.4 that certain patterns exist in the head pose distribution - close points turn to cluster together. That observation occurs for other sequences as well. As a result, it is wanted to select keyframes out of a video by clustering the 3D head poses. The widely-used K-means clustering aims to partition the point set into K subsets so as to minimize the within-cluster Sum of Squared Distances (SSD). If each cluster is treated as a class, the model minimizes the intra-class or within-cluster distance.

Second, the pose representation is representative in terms of diversity (*i.e.*, dif-

ference, distance). Intuitively, it is wanted to retain the key faces that have poses as different as possible. If each frame's estimated 3D pose are treated as a point, then the approximate polygon formed by selected points should be as close to the true polygon formed by all the points as possible. The diversity is measured using the SSD between any two selected key points (SSD within the set formed by centroids if they are used as key points). And it is wanted to maximize such an inter-class or between-cluster distance.

Now, the criteria are put together in a single objective. There are a limited number of choices for K as its upper bound is set. Then, given a set $\mathbb{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\}$ of pose observations, it is aimed to partition the m observations into K ($\leq m$) **disjoint** subsets $\mathbb{S} = \{\mathbb{S}_1, \mathbb{S}_2, ..., \mathbb{S}_K\}$ so as to minimize the within-cluster SSD as well as maximize the between-cluster SSD while still minimizing the number of clusters:

$$\min_{K,\mathbb{S}} \frac{SSD_{within}}{SSD_{between}} := \sum_{k=1}^{K} \frac{\sum_{i=1}^{m} \|\mathbf{p}_i - \mu_k\|^2}{\sum_{j=1, j\neq k}^{K} \|\mu_j - \mu_k\|^2} \tag{3.1}$$

where μ_j, μ_k is the mean of points in $\mathbb{S}_k, \mathbb{S}_k$, respectively. This objective differs from that of K-means only in considering between-cluster distance which makes it a bit similar with multi-class LDA (Linear Discriminant Analysis). However, it is still essentially K-means. To solve it, the alternative minimization are not really needed because that K with a limited number of choices is empirically enumerated by cross validation. Once K is fixed, solving Eqn. 3.1 follows a similar procedure of multi-class LDA while there is no mixture of classes or clusters because every point is hard-assigned to a single cluster as done in K-means. Theoretically, it can be proven

that the pose diversity (induced by Eqn. 3.1's solution) $\sum_{k=1}^{K}\sum_{j=1,j\neq k}^{K}\|\mu_j-\mu_k\|^2 \geq \sum_{k=1}^{K}\sum_{j=1,j\neq k}^{K}\|\mathbf{r}_j-\mathbf{r}_k\|^2$ where $\mathbf{r}_k$ is K randomly sampled pose points which surely include the case given by the standard K-means $\min_{\mathbb{S}}\sum_{k=1}^{K}\sum_{i=1}^{m}\|\mathbf{p}_i-\mu_k\|^2$. The subsequent selection of key poses is straightforward (by the distances to K-means centroids). The selected key poses form a subset $\mathbb{P}_\Omega$ of $\mathbb{P}$ where $\mathbf{\Omega}$ is an m-dimensional K-**sparse** impulse vector of binary values 1/0 indicating whether the index is chosen or not, respectively.

The selection of frames will follow the index activation vector $\mathbf{\Omega}$ as well. Such a selection reduces the number of images required to represent the face from tens or hundreds to K while preserving the pose diversity which is considered in the formation of clusters. Now the chosen faces are frontalized, which is called face alignment or pose correction/normalization. All above operations are summarized in Algorithm 1. Note that not all landmarks can be perfectly aligned. Priority is given to salient ones such as the eye center and corners, the nose tip, the mouth corners and the chin. Other properties such as symmetry are also preserved. For example, the detected eye is mirrored horizontally. However, a profile will not be frontalized.

3.1.3 Pooling max correlation as similarity

This section explains our max correlation guided pooling from a set of deep face features and verify whether the selected keyframes are able to well represent identity regardless of pose variation.

Algorithm 1: K frame selection.

Input : face video $\mathbb{V} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_m\}$.

Output: pose-corrected down-sampled face video $\mathbb{V}_\Omega^c = \{\mathbf{x}_{(1)}^c, \mathbf{x}_{(2)}^c, ..., \mathbf{x}_{(K)}^c\}$.

(1) Landmark detection: detect facial landmarks per frame in $\mathbb{V}$ so that correspondence between frames is known.

(2) Homography estimation: estimate an approximate 3D model (say, homography $\mathbf{H}$) from the sequence of faces in $\mathbb{V}$ with known correspondence from landmarks.

(3) Pose estimation: compute the rotation angles p_i for each frame using landmark correspondence and obtain a set of sequential head poses $\mathbb{P} = \{\mathbf{p}_1, \mathbf{p}_2, ..., \mathbf{p}_m\}$.

(4) Pose quantization: cluster $\mathbb{P}$ into K subsets $\mathbb{S}_1, \mathbb{S}_2, ..., \mathbb{S}_K$ by solving Eqn. 3.1 with estimated pose centroids $\{\mathbf{c}_1, ..., \mathbf{c}_K\}$ which might be pseudo pose (non-existing pose).

(5) Pose selection: for each cluster, compute the distances from each pose point $\mathbf{p} \in \mathbb{S}_k$ to the pose centroid c_k and then select the closest pose point to represent the cluster $\mathbb{S}_k$. The selected key poses form a subset $\mathbb{P}_{\mathbf{\Omega}}$ of $\mathbb{P}$ where $\mathbf{\Omega}$ is the index activation vector.

(6) Face selection: follow $\mathbf{\Omega}$ to select the keyframes and form a subset $\mathbb{V}_\Omega = \{\mathbf{x}_{(1)}, \mathbf{x}_{(2)}, ..., \mathbf{x}_{(K)}\}$ of $\mathbb{V}$ where $\mathbb{V}_\Omega \subset \mathbb{V}$.

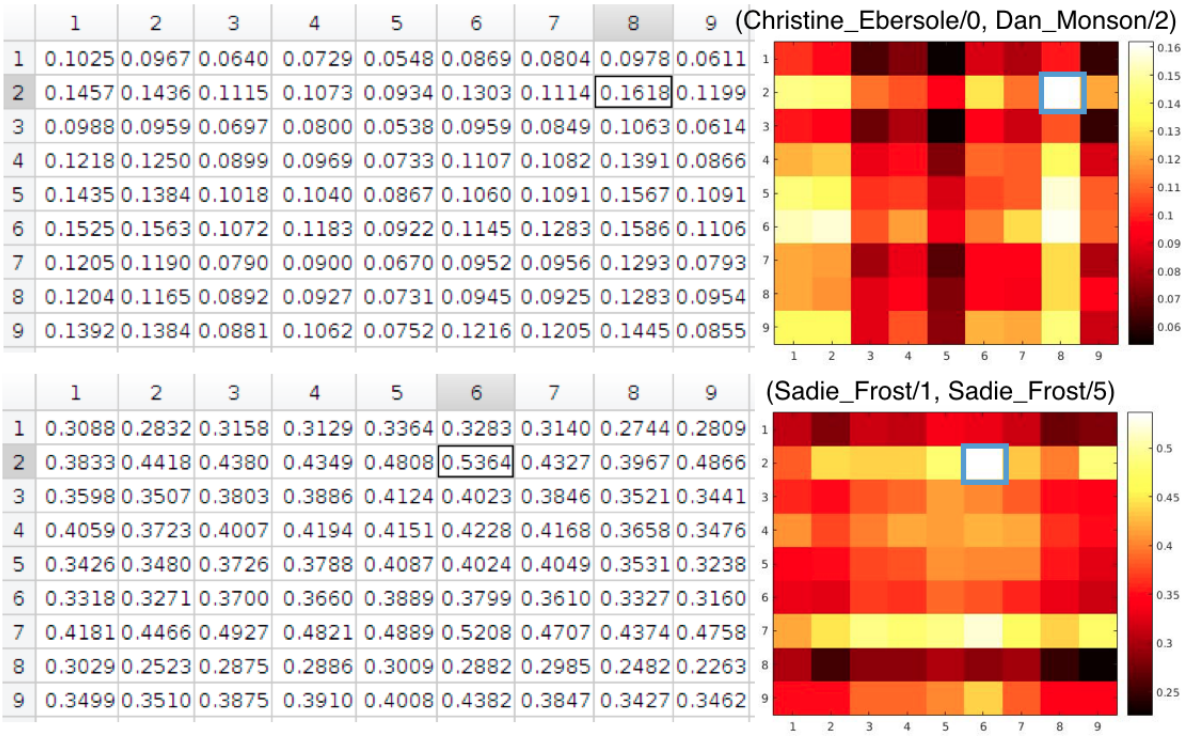
(7) Face alignment: Warp each face in $\mathbb{V}_\Omega$ according to $\mathbf{H}$ so that landmarks are fixed to canonical positions.

After face alignment, some feature descriptor, a function $f(\cdot)$, maps each corrected frame $\mathbf{x}^c_{(i)}$ to a $d \times 1$ feature vector $f(\mathbf{x}^c_{(i)}) \in \mathbb{R}^d$ with dimensionality d and unit Euclidean norm. Then the video is represented as a bag of normalized frame-wise CNN features $\mathbb{X} := \{\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_K\} := \{f(\mathbf{x}^c_{(1)}), f(\mathbf{x}^c_{(2)}), ..., f(\mathbf{x}^c_{(K)})\}$. The feature vectors can also be arranged column by column to form a matrix $\mathbf{X} = \left[\mathbf{f}_1|\mathbf{f}_2|...|\mathbf{f}_K\right]$. For example, the VGG-face network [99] has been verified to be able to produce features well representing the identity information. It has 24 layers including several stacked convolution-pooling layer, 2 fully-connected layer and one softmax layer. Since the model was trained for face identification purpose with respect to 2,622 identities, the output of the second last fully-connected layer is used as the feature descriptor, which returns a 4,096-dim feature vector for each input face.

Given a pair of videos $(\mathbb{V}_a, \mathbb{V}_b)$ of the subject a and b respectively, it is wanted to measure the similarity between a and b. Since it is claimed that the proposed bag of CNN features can well represent the identity, instead the model measures the similarity between two sets of CNN features $Sim(\mathbb{X}_a, \mathbb{X}_b)$ which is defined as the max correlation among all possible pairs of CNN features, namely the max element in the correlation matrix (see Fig. 3.5):

$$Sim(\mathbb{X}_a, \mathbb{X}_b) := \max_{n_a, n_b}(\mathbf{f}^{a\ T}_{n_a} \cdot \mathbf{f}^b_{n_b}) = \max\left((\mathbf{X}_a{}^T \mathbf{X}_a)(:)\right) \tag{3.2}$$

where $n_a = 1, 2, ..., K_a$ and $n_b = 1, 2, ..., K_b$. Notably, the notation (:) indicates all elements in a matrix following the MATLAB convention. Now, instead of comparing



	1	2	3	4	5	6	7	8	9
1	0.1025	0.0967	0.0640	0.0729	0.0548	0.0869	0.0804	0.0978	0.0611
2	0.1457	0.1436	0.1115	0.1073	0.0934	0.1303	0.1114	0.1618	0.1199
3	0.0988	0.0959	0.0697	0.0800	0.0538	0.0959	0.0849	0.1063	0.0614
4	0.1218	0.1250	0.0899	0.0969	0.0733	0.1107	0.1082	0.1391	0.0866
5	0.1435	0.1384	0.1018	0.1040	0.0867	0.1060	0.1091	0.1567	0.1091
6	0.1525	0.1563	0.1072	0.1183	0.0922	0.1145	0.1283	0.1586	0.1106
7	0.1205	0.1190	0.0790	0.0900	0.0670	0.0952	0.0956	0.1293	0.0793
8	0.1204	0.1165	0.0892	0.0927	0.0731	0.0945	0.0925	0.1283	0.0954
9	0.1392	0.1384	0.0881	0.1062	0.0752	0.1216	0.1205	0.1445	0.0855

	1	2	3	4	5	6	7	8	9
1	0.3088	0.2832	0.3158	0.3129	0.3364	0.3283	0.3140	0.2744	0.2809
2	0.3833	0.4418	0.4380	0.4349	0.4808	0.5364	0.4327	0.3967	0.4866
3	0.3598	0.3507	0.3803	0.3886	0.4124	0.4023	0.3846	0.3521	0.3441
4	0.4059	0.3723	0.4007	0.4194	0.4151	0.4228	0.4168	0.3658	0.3476
5	0.3426	0.3480	0.3726	0.3788	0.4087	0.4024	0.4049	0.3531	0.3238
6	0.3318	0.3271	0.3700	0.3660	0.3889	0.3799	0.3610	0.3327	0.3160
7	0.4181	0.4466	0.4927	0.4821	0.4889	0.5208	0.4707	0.4374	0.4758
8	0.3029	0.2523	0.2875	0.2886	0.3009	0.2882	0.2985	0.2482	0.2263
9	0.3499	0.3510	0.3875	0.3910	0.4008	0.4382	0.3847	0.3427	0.3462

Figure 3.5: Max pooling from the correlation matrix with each axis coordinates the time step in one video. Top row gives an example of different subjects while the bottom row shows that of the same person. Max responses are highlighted by boxes. Faces not shown due to copyright consideration.

$m_a \times m_b$ pairs, with Sec. 3.1.2 it is only needed to compute $K_a \times K_b$ correlations, from which the model further pools a single (1×1) number as the similarity measure. In the time domain, it also serves as pushing from K images to just 1 image. The metric can be the mean, median, max or the majority of a histogram while the mean and max are more widely-used. The insight of not taking the mean is that a frame highly correlated with another video usually does not appear twice in a temporal sliding window. If the two bags of features are plotted in the common feature space, a similarity is essentially the closeness between the two sets of points. If the two

sets are non-overlapping, one measure of the closeness between two points sets is the distance between nearest neighbors, which is essentially pooling the max correlation. Similar to spatial pooling for invariance, taking the max from the correlation matrix shown in Fig. 3.5 preserves the temporal invariance that the largest correlation can appear at any time step among the selected frames. Since the identity is consistent in one video, it can be claimed two videos contain a similar person as long as one pair of frames from each video are highly correlated. The computation of two videos' identity similarity is summarized in Algorithm 2.

Algorithm 2: Video-based identity similarity measurement.

Input : A pair of face videos $\mathbb{V}_a$ and $\mathbb{V}_b$.

Output: The similarity score $Sim(\mathbb{X}_a, \mathbb{X}_b)$ of their subject identity.

(1) Face selection and alignment: run Algorithm 1 for each video to obtain keyframes with faces aligned.

(2) Deep video representation: generate deep face features of the keyframes to obtain two sets of features $\mathbb{X}_a$ and $\mathbb{X}_b$.

(3) Pooling max correlation: compute similarity $Sim(\mathbb{X}_a, \mathbb{X}_b)$ according to Eqn. 3.2.

3.1.4 Experiments

In the previous section, an algorithm for face verification in videos has been proposed. Specifically, an algorithm for selecting K frames from a video of one person

is presented. It allows preserving pose diversity and reduces the number of images required to represent a face from a video. The selected frames are used to compute the identity similarity between two sets of deep face features obtained with VGG-face net, by using max correlation. In the following, this section presents the details about implementation (Sec. 6.2.6.1) and the experiment on YTF (Sec. 3.1.4.2) with an analysis of the comparable results with VGG-face.

3.1.4.1 Implementation

Programs are developed using resources[1] such as OpenCV, DLib and VGG-Face.

- Face detection: frame-by-frame detection using DLib's HOG+SVM based detector trained on 3,000 cropped face images from LFW. It works better for faces in the wild than OpenCV's cascaded haar-like+boosting based detector.
- Facial landmark: DLib's landmark model trained via regression tree ensemble.
- Head pose estimation: OpenCV's solvePnP recovering 3D coordinates from 2D coordinates via Direct Linear Transform + Levenberg-Marquardt optimization.
- Face alignment: warpAffine by affine-warping to center eyes and mouth.
- Deep face representation [2]: second last layer output (4,096-dim) of VGG-Face [99] using Caffe [102]. It has been trained using face images of size 224 ×

[1] `http://opencv.org/`, `http://dlib.net/` and `http://www.robots.ox.ac.uk/~vgg/software/vgg_face/`, respectively.

[2] Codes are available at `https://github.com/eglxiang/vgg_face`

224 with the average face image subtracted and then is used for our verification purpose without any re-training. However, such average face subtraction is unavailable and unnecessary given a new inputting image. Notably, face images are directly inputted to the VGG-Face network without any mean face subtraction.

3.1.4.2 Evaluation on video-based face verification

For video-based face recognition database, EPFL captures 152 people facing Webcam and mobile-phone camera in controlled environments. However, they are frontal faces and thus of no use to us. University of Surrey and University of Queensland capture 295 and 45 subjects under various well-quantized poses in controlled environments, respectively. Since the poses are well quantized, our pose quantization and selection algorithm can hardly be verified on them. McGill and NICTA capture 60 videos of 60 subjects and 48 surveillance videos of 29 subjects in uncontrolled environments, respectively. However, the database size is way too small. YouTube Faces (YTF) dataset (YTF) and India Movie Face Database (IMFDB) collect 3,425 videos of 1,595 people and 100 videos of 100 actors in uncontrolled environments, respectively. There is quite a few existing works verified on IMFDB. As a result, the YTF dataset [3] [95] is chosen to verify the proposed video-based similarity measure for face verification. YTF was built by using the 5,749 names of subjects included in the

[3]Dataset is available at `http://www.cs.tau.ac.il/~wolf/ytfaces/`

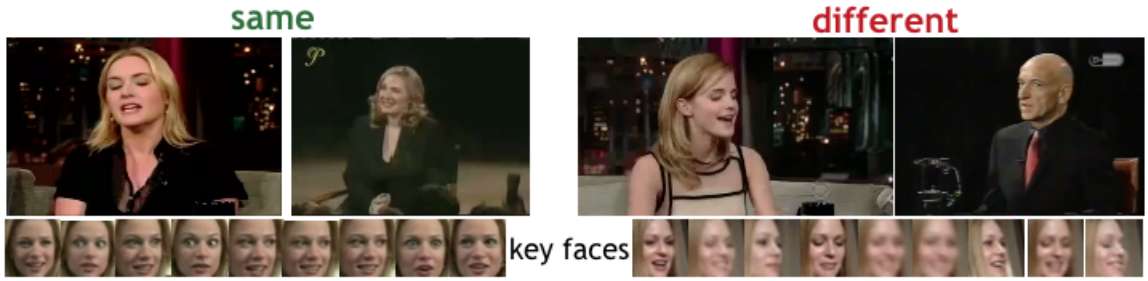


Figure 3.6: Examples of YFT video-pairs. Instead of using the full video in the top row, key faces in the bottom row are chosen.

LFW dataset [96] to search YouTube for videos of these same individuals. Then, a screening process reduced the original set of videos from the 18,899 of 3,345 subjects to 3,425 videos of 1,595 subjects.

In the same way with LFW, the creator of YTF provides an initial official list of 5,000 video pairs with ground truth (same person or not as shown in Fig. 3.6). Our experiments can be replicated by following our tutorial [4]. $K = 9$ turns to be averagely the best for the YTF dataset. Fig. 3.7 presents the Receiver Operating Characteristic (ROC) curve obtained after the 5,000 video-video similarity scores are computed. One way to look at a ROC curve is to first fix the level of false positive rate that can be allowed (say, 0.1) and then see how high is the true positive rate (say, roughly 0.9). Another way is to see how close the curve towards the top-left corner. Namely, the Area Under the Curve (AUC) is measured and hoped to be as large as possible. In this testing, **the AUC is 0.9419 which is quite close to VGG-Face [99] which uses temporal mean pooling**. However, our selective

[4]Codes with a tutorial at `https://github.com/eglxiang/ytf`

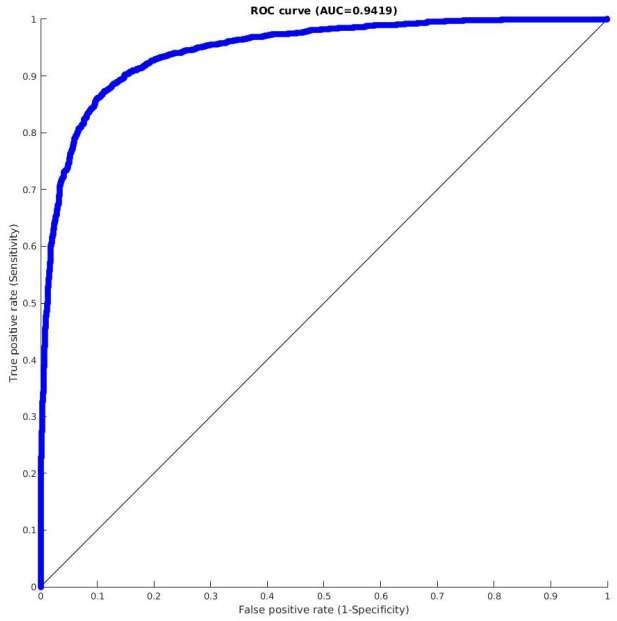


Figure 3.7: ROC curve of running our algorithm on the YTF initial official list of 5,000 pairs.

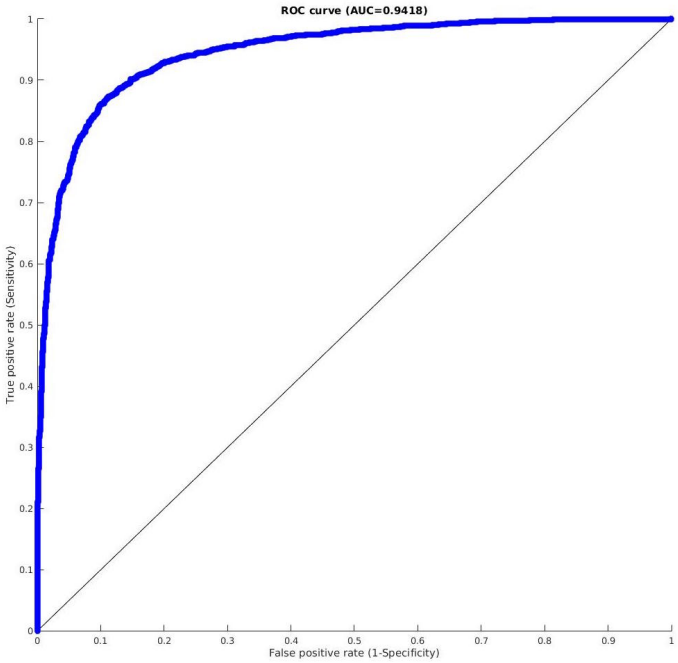


Figure 3.8: ROC curve of running our algorithm on the YTF corrected official list of 4,999 video pairs.

pooling strategy has much fewer computation credited to the key face selection which also preserves the pose diversity. Notably, alternative strategies exist for keyframe selection, such as ordering the confidence of the facial landmarks detection used in VGG-face which does not give any analysis why the video down-sampling does not hurt the recognition. Instead, our proposed down-sampling strategy is technically sounded in the sense of persevering pose diversity. Later on, the creator of YTF sends a list of errors in the ground-truth label file and provides a corrected list of video pairs with updated ground-truth labels. As a result, the proposed algorithm is tested again on the corrected 4,999 video pairs. Fig. 3.8 updates the ROC curve with an AUC of 0.9418 which is identical with the result on the initial list.

3.1.5 Summary

This section has proposed a K frame selection algorithm and an identity similarity measure which employs simple correlations and no learning. It is verified on fast video-based face verification on YTF and achieves comparable performance with VGG-face. Particularly, the selection and pooling significantly reduce the computational expense of processing videos. The further verification of the proposed algorithm includes the evaluation of video-based face expression recognition. A generic problem underneath is variable disentanglement in real data and a take-home message is that employing geometric cues can improve the descriptiveness of deep features.

3.2 Regularizing normalized FaceNet for facial action quality assessment

This section takes the example of facial pain intensity estimation as an extreme case of image set representation. As previously mentioned, a video can be simply treated as a set of data points $(frame, intensity)$. It is aimed at learning the mapping from $frame$ to $intensity$. The data points across different training videos are mixed to form a training set. Essentially, what are explained in the following is image-based learning, instead of video-based learning where all data points from the same video might be treated as a single training sample.

However, even though there seem a huge number of data points, a practical concern is that there are actually not too many videos labeled with pain intensities. The small available dataset prevents us from directly training a deep pain intensity regressor. In the following, it is shown that fine-tuning from a data-extensive pre-trained domain such as face verification can alleviate this problem. Our solutions are

• fine-tuning a well-trained face verification net on additional data with a regularized regression loss and a hidden full-connected layer regularized using dropout,

• regularizing the regression loss using a center loss,

• and re-sampling the training data by the population proportion of a certain pain intensity *w.r.t.* the total population.

While our work is not the first attempt of this regularization idea [103], to our

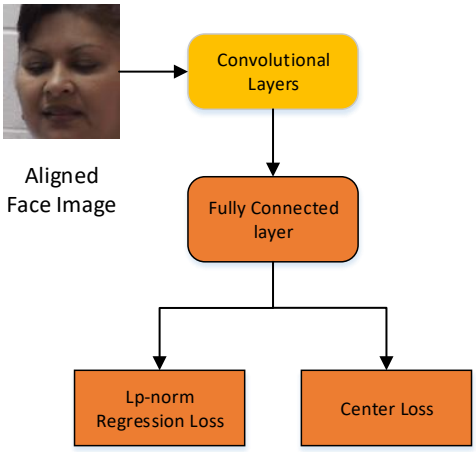


Figure 3.9: Simplified illustration of the network architecture. The convolution layers are adapted from the NormFace. There are two new FC layers. To avoid over-fitting the limited data, the number of neurons in our hidden FC layer is relatively smaller than the previous layer (50 vs 512), known as Dropout [6] as regularization.

knowledge it is the first time to apply that idea to the pain expression intensity estimation. Correspondingly, three solutions are proposed to address the four issues mentioned above. In summary, the contributions of this work include

• addressing limited data with expression intensity labels by relating two mappings from the same input face space to different output label space where the identity labels are rich,

• pushing the pain assessment performance by a large margin,

• proposing to add center loss regularizer to make the regressed values closer to discrete values,

• and proposing a more sensible evaluation metric to address the imbalance issue caused by the phenomena that most of the time a patient does not express pain.

Our network is based on our face verification network [104][5] trained using the CASIA-WebFace dataset containing 0.5 million face images with identity labels. As a classification network, it employs the Softmax loss regularized with its proposed center loss. But it is difficult to directly fine-tune the network for pain intensity classification due to limited face images with pain labels. However, it is feasible to fit the data points $(feature, intensity)$ as a regression problem. Our fine-tuning network employs a regression loss regularized with the center loss, as shown in Fig. 3.9.

Firstly, the face verification net's softmax loss is modified to be a Mean Square Error (MSE) loss for regression. The last layer of such a network is a $\ell 2$ distance layer, which easily causes gradient exploding due to large magnitudes of the gradients at initial iterations. Thus, the MSE loss is replaced using a smooth $\ell 1$ loss with a Huber loss flavor (see Sec. 3.2.3).

Secondly, as labels are discrete, it is sensible to regularize the loss to make the regressed values to be more discrete. The center loss [105] is introduced as a regularizer (see Sec. 4.1.5).

Thirdly, two weighted evaluation metrics are proposed in Sec.3.2.5.2 to address label imbalance which may induce trivial method. In the following, three solutions are given.

[5]Model available at `https://github.com/eglxiang/NormFace`

3.2.1 Deep face networks related works

Two pieces of recent work make progress in estimating pain intensity visually using the Shoulder-Pain dataset only: Ordinal Support Vector Regression (OSVR) [106] and Recurrent Convolutional Regression (RCR) [107]. Notably, RCR [107] is trained end-to-end yet achieving sub-optimal performance. Please see reference therein for other existing works. For facial expression recognition in general, there is a trade-off between method simplicity and performance, *i.e.*, image-based [103, 108] *vs.* video-based [25, 109–111] methods. As videos are sequential signals, appearance-based methods including ours cannot model the dynamics given by a temporal model [109] or spatiotemporal models [25, 110, 111]. Linear models include sparse representation based method, ordinal regression [106, 112, 113] and boosting [114]. Similar tradeoff also lies in linear model *vs.* non-linear models. Among non-linear models, one approach is kernel-based methods [115] while another is deep learning [107, 116–119]. By introducing more information, one approach is 3D models [120] while another is multi-modal models [121].

Normalization is a common operation in modern neural network models. Local Response Normalization and Local Contrast Normalization are studied in the AlexNet model [122], even though these techniques are no longer common in modern models. Batch normalization [123] is widely used to accelerate the speed of neural network convergence by reducing the internal covariate shift of intermediate features. Weight normalization [124] was proposed to normalize the weights of convolution layers and

inner-product layers, and also lead to faster convergence speed. Layer normalization [125] tried to solve the batch size dependent problem of batch normalization, and works well on Recurrent Neural Networks. Recently, cosine similarity [126] was used instead of the inner-product for training a CNN for person recognition, which is quite similar to face verification. The Cosine Loss proposed in [126] is quite similar with the one described in Section 3.2.2.3, normalizing both the features and weights. L2-Softmax [127] shares a similar analysis about the convergence problem described in Section 3.2.2.3. In [127], the authors also propose to add a scale parameter after normalization, but they only normalize the features. SphereFace [128] improves the performance of Large Margin Softmax [129] by normalizing the weights of the last inner-product layer only. Von Mises-Fisher Mixture Model(vMFMM) [130] interprets the hypersphere embedding as a mixture of von Mises-Fisher distributions. To sum up, the Cosine Loss [126], vMFMM [130] and our proposed loss functions optimize both features and weights, while the L2-Softmax [127] normalizes the features only and the SphereFace [128] normalizes the weights only.

3.2.2 Normalization layer for facial recognition

Thanks to the recent developments of Convolutional Neural Networks, the performance of face verification methods has increased rapidly. In a typical face verification method, feature normalization is a critical step for boosting performance. This motivates us to introduce and study the effect of normalization during training. But it is

known that this is non-trivial, despite normalization being differentiable. Four issues have been identified related to normalization through mathematical analysis, which yields understanding and helps with parameter settings. Based on this analysis, two strategies for training using normalized features are proposed. The first is a modification of the Softmax loss, which optimizes cosine similarity instead of inner-product. It is shown that the strategy consistently improves performance by between 0.2% to 0.4% on the Labeled Face in the Wild (LFW) dataset based on two models. This is significant because the performance of the two models on the LFW dataset is close to saturation at over 98%. In this section, questions are answered regarding why to normalize the features when the loss function is the Softmax loss and why the network does not converge if the Softmax loss is not directly put on the normalized features.

3.2.2.1 Necessity of Normalization

In order to give an intuitive feeling about the Softmax loss, a toy experiment of training a deeper LeNet [131] model is done on the MNIST dataset [7]. The number of feature dimension has been reduced to 2. The 10,000 2D features from the training set have been plotted on a 2D plane in Figure 3.10. From the figure, it can be found that $\mathbf{f_2}$ can be much closer to $\mathbf{f_1}$ than to $\mathbf{f_3}$ if the Euclidean distance is used as the metric. Hence, directly using the features for comparison may lead to a bad performance. At the same time, it can be found that the angles between feature vectors seem to be a good metric compared with Euclidean distance or inner-product operations.

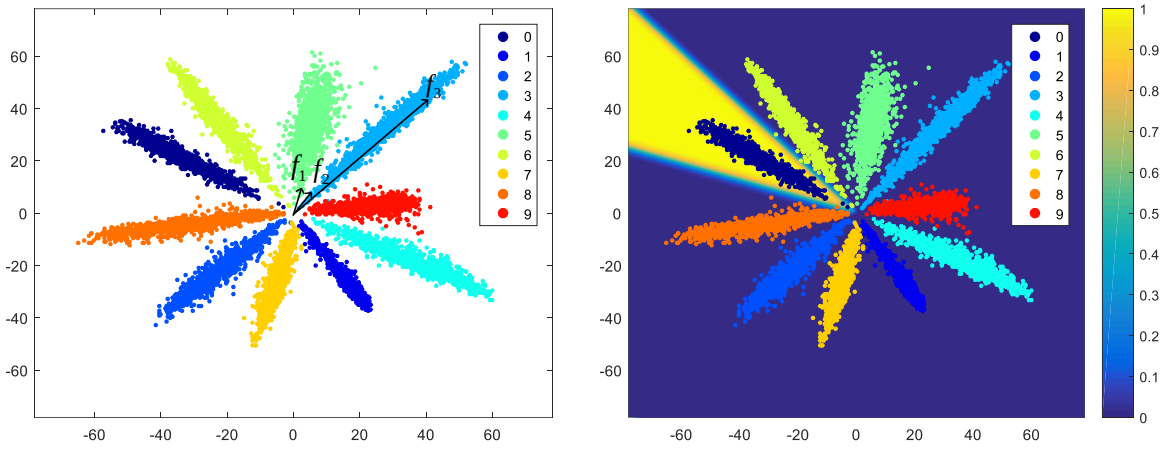


Figure 3.10: *Left:* The optimized 2-dimensional feature distribution using Softmax loss on MNIST [7] dataset. Note that the Euclidean distance between $\mathbf{f_1}$ and $\mathbf{f_2}$ is much smaller than the distance between $\mathbf{f_2}$ and $\mathbf{f_3}$, even though $\mathbf{f_2}$ and $\mathbf{f_3}$ are from the same class. *Right:* The softmax probability for class 0 on the 2-dimension plane. Best viewed in color.

Actually, most previous work takes the cosine of the angle between feature vectors as the similarity [105, 132, 133], even though they all use the Softmax loss to train the network. Since the most common similarity metric for the Softmax loss is the inner-product with unnormalized features, there is a gap between the metrics used in the training and testing phases.

The reason why the Softmax loss tends to create a 'radial' feature distribution (Figure 3.10) is that the Softmax loss actually acts as the *soft* version of the *max* operator. Scaling the feature vectors' magnitude does not affect the assignment of its class. Formally speaking, the definition of the Softmax loss is as followed.

$$\mathcal{L}_S = -\frac{1}{m}\sum_{i=1}^{m} log\frac{e^{W_{y_i}^T \mathbf{f}_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j^T \mathbf{f}_i + b_j}}, \tag{3.3}$$

where m is the number of training samples, n is the number of classes, $\mathbf{f}_i$ is the feature

of the i-th sample, y_i is the corresponding label in range $[1, n]$, W and b are the weight matrix and the bias vector of the last inner-product layer before the Softmax loss, W_j is the j-th column of W, which is corresponding to the j-th class. In the testing phase, a sample is classified by

$$Class(\mathbf{f}) = i = \arg\max_i (W_i^T \mathbf{f} + b_i). \tag{3.4}$$

In this case, it can be inferred that $(W_i\mathbf{f} + b_i) - (W_j\mathbf{f} + b_j) \geq 0, \forall j \in [1, n]$. Using this inequality, the following proposition is obtained.

Proposition 1. *For the softmax loss with* ***no-bias*** *inner-product similarity as its metric, let* $P_i(\mathbf{f}) = \frac{e^{W_i^T \mathbf{f}}}{\sum_{j=1}^n e^{W_j^T \mathbf{f}}}$ *denote the probability of* $\mathbf{x}$ *being classified as class* i*. For any given scale* $s > 1$*, if* $i = \arg\max_j (W_j^T \mathbf{f})$*, then* $P_i(s\mathbf{f}) \geq P_i(\mathbf{f})$ *always holds.*

Proof: Let $t = s - 1$, after scaling, there is,

$$\begin{aligned} P_i(s\mathbf{f}) &= \frac{e^{W_i^T[(1+t)\mathbf{f}]}}{\sum_{j=1}^n e^{W_j^T[(1+t)\mathbf{f}]}} \\ &= \frac{e^{W_i^T \mathbf{f}}}{\sum_{j=1}^n e^{W_j^T \mathbf{f} + t(W_j^T \mathbf{f} - W_i^T \mathbf{f})}}. \end{aligned} \tag{3.5}$$

Recall that $W_i\mathbf{f} - W_j\mathbf{f} \geq 0$ if $i = \arg\max_j (W_j\mathbf{f})$, so $t(W_j^T\mathbf{f} - W_i^T\mathbf{f}) \leq 0$ always holds. Then

$$\begin{aligned} P_i(s\mathbf{f}) &\geq \frac{e^{W_i^T \mathbf{f}}}{\sum_{j=1}^n e^{W_j^T \mathbf{f}}} \\ &= P_i(\mathbf{f}). \end{aligned} \tag{3.6}$$

The equality holds if $W^T\mathbf{f} = \mathbf{0}$ or $W_i = W_j, \forall i, j \in [1, n]$, which is almost impossible in practice.

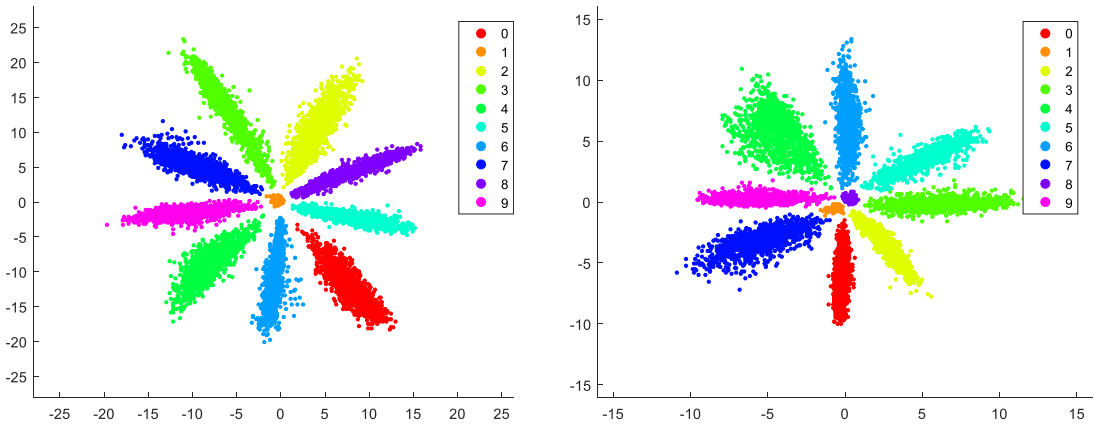


Figure 3.11: Two selected scatter diagrams when bias term is added after inner-product operation. Please note that there are one or two clusters that are located near the zero point. If the features of the center clusters are normalized, they would spread everywhere on the unit circle, which would cause misclassification. Best viewed in color.

This proposition implies that softmax loss always encourages well-separated features to have bigger magnitudes. This is the reason why the feature distribution of softmax is 'radial'. However, this property may not be needed as shown in Fig. 3.10. By normalization, its effect can be eliminated. Thus, the cosine of two feature vectors is usually used to measure the similarity of two samples.

However, the Proposition 1 does not hold if a bias term is added after the inner-product operation. In fact, the weight vector of the two classes could be the same and the model still could make a decision via the biases. It could be found this kind of case during the MNIST experiments and the scatters are shown in Figure 3.11. It can be discovered from the figure that the points of some classes all are located around the zero point. After normalization the points from each of these classes may

be spread out on the unit circle, overlapping with other classes. In these cases, feature normalization may destroy the discrimination ability of the specific classes. To avoid this kind of risk, the bias term is not added before the Softmax loss in this work, even though it is commonly used for classification tasks.

3.2.2.2 Layer Definition

In this section, it can be defined that $\|\mathbf{x}\|_2 = \sqrt{\sum_i \mathbf{x}_i^2 + \epsilon}$, where ϵ is a small positive value to prevent dividing zero. For an input vector $\mathbf{x} \in \mathcal{R}^n$, an L_2 normalization layer outputs the normalized vector,

$$\tilde{\mathbf{x}} = \frac{\mathbf{x}}{\|\mathbf{x}\|_2} = \frac{\mathbf{x}}{\sqrt{\sum_i \mathbf{x}_i^2 + \epsilon}}. \tag{3.7}$$

Here $\mathbf{x}$ can be either the feature vector $\mathbf{f}$ or one column of the weight matrix W_i. In backward propagation, the gradient w.r.t. $\mathbf{x}$ can be obtained by the chain-rule,

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mathbf{x}_i} &= \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}_i}\frac{\partial \tilde{\mathbf{x}}_i}{\partial \mathbf{x}_i} + \sum_j \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}_j}\frac{\partial \tilde{\mathbf{x}}_j}{\partial \|\mathbf{x}\|_2}\frac{\partial \|\mathbf{x}\|_2}{\partial \mathbf{x}_i} \\ &= \frac{\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}_i} - \tilde{\mathbf{x}}_i \sum_j \frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}_j}\tilde{\mathbf{x}}_j}{\|\mathbf{x}\|_2}. \end{aligned} \tag{3.8}$$

It is noteworthy that vector $\mathbf{x}$ and $\frac{\partial \mathcal{L}}{\partial \mathbf{x}}$ are orthogonal with each other, *i.e.* $\langle \mathbf{x}, \frac{\partial \mathcal{L}}{\partial \mathbf{x}} \rangle = 0$. From a geometric perspective, the gradient $\frac{\partial \mathcal{L}}{\partial \mathbf{x}}$ is the projection of $\frac{\partial \mathcal{L}}{\partial \tilde{\mathbf{x}}}$ onto the tangent space of the unit hypersphere at normal vector $\tilde{x}$ (see Figure 3.12). From Figure 3.12 left, it can be inferred that after update, $\|\mathbf{x}\|_2$ always increases. In order to prevent $\|\mathbf{x}\|_2$ growing infinitely, weight decay is necessary on vector $\mathbf{x}$.

3.2.2.3 Reformulating Softmax Loss

Using the normalization layer, the model directly optimizes the cosine similarity,

$$d(\mathbf{f}, \mathbf{W_i}) = \frac{\langle \mathbf{f}, \mathbf{W_i} \rangle}{\|\mathbf{f}\|_2 \|\mathbf{W_i}\|_2}, \tag{3.9}$$

where $\mathbf{f}$ is the feature and $\mathbf{W_i}$ represents the i-th column of the weight matrix of the inner-product layer before the Softmax loss layer. However, after normalization, the network fails to converge. The loss only decreases a little and then converges to a very big value within a few thousands of iterations. After that the loss does not decrease no matter how many iterations the training has and how small the learning rate is.

This is mainly because the range of $d(\mathbf{f}, \mathbf{W_i})$ is only $[-1, 1]$ after normalization, while it is usually between $(-20, 20)$ and $(-80, 80)$ when an inner-product layer and softmax loss are used. This low range problem may prevent the probability $P_{y_i}(\mathbf{f}; \mathbf{W}) = \frac{e^{\mathbf{W_{y_i}^T f}}}{\sum_j^n e^{\mathbf{W_j^T f}}}$, where y_i is $\mathbf{f}$'s label, from getting close to 1 even when the samples are well-separated. In the extreme case, $\frac{e^1}{e^1+(n-1)e^{-1}}$ is very small (0.45 when $n = 10$; 0.007 when $n = 1000$), even though in this condition the samples of all other classes are on the other side of the unit hypersphere. Since the gradient of softmax loss w.r.t. the ground truth label is $1 - P_{y_i}$, the model will always try to give large gradients to the well-separated samples, while the harder samples may not get sufficient gradients. To better understand this problem, a bound is now given to clarify how small the Softmax loss can be in the best case.

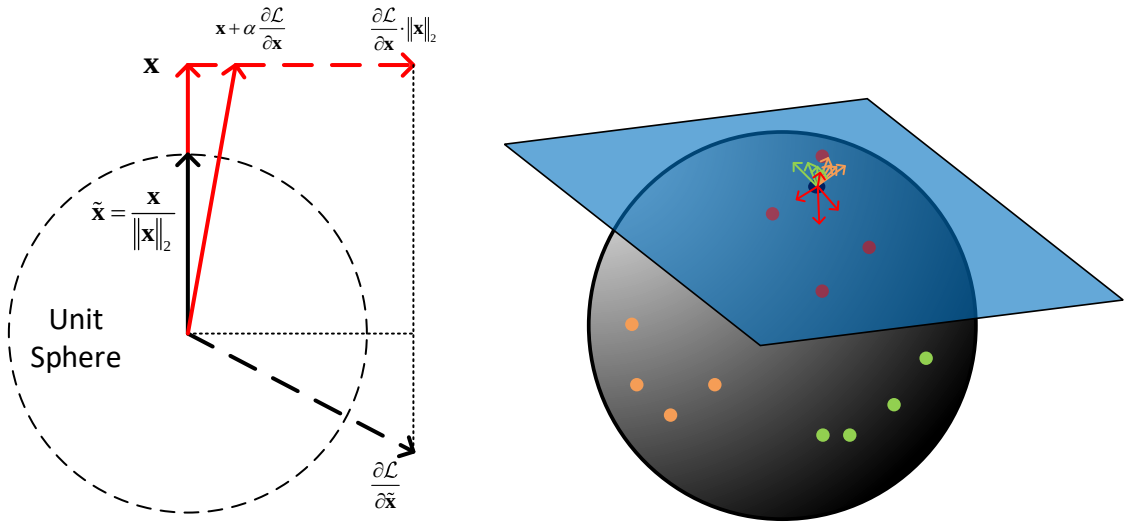


Figure 3.12: *Left:* The normalization operation and its gradient in 2-dimensional space. Please note that $\|\mathbf{x}+\alpha\frac{\partial\mathcal{L}}{\partial\mathbf{x}}\|$ is always bigger than $\|\mathbf{x}\|$ for all $\alpha > 0$ because of the Pythagoras theorem. *Right:* An example of the gradients w.r.t. the weight vector. All the gradients are in the tangent space of the unit sphere (denoted as the blue plane). The red, yellow and green points are normalized features from 3 different classes. The blue point is the normalized weight corresponding to the red class. Here it is assumed that the model tries to make features get close to their corresponding classes and away from other classes. Even though the illustration is for the gradients applied on the normalized weight only, please note that opposite gradients are also applied on the normalized features (red, yellow, green points). Finally, all the gradients are accumulated together to decide which direction the weight should be updated. Best viewed in color.

Proposition 2. (Softmax Loss Bound After Normalization) *Assume that every class has the same number of samples, and all the samples are well-separated, i.e., each sample's feature is exactly same with its corresponding class's weight. If the model normalizes both the features and every column of the weights to have a norm of ℓ, the softmax loss will have a lower bound,* $\log\left(1+(n-1)\,e^{-\frac{n}{n-1}\ell^2}\right)$*, where n is the class number.*

Proof: Assume $\|W_i\| = \ell, \forall i \in [1, n]$ for convenience. Since it is already assumed

that all samples are well-separated, W_i is directly used to represent the i-th class' feature. The definition of the Softmax loss is,

$$\mathcal{L}_{\mathcal{S}} = -\frac{1}{n}\sum_{i=1}^{n}\log\frac{e^{W_i^T W_i}}{\sum_{j=1}^{n} e^{W_i^T W_j}}. \tag{3.10}$$

This formula is different from Equation (3.3) because it is assumed that every class has the same sample number. By dividing $e^{W_i^T W_i} = e^{\ell^2}$ from both the numerator and denominator,

$$\begin{aligned}\mathcal{L}_{\mathcal{S}} &= -\frac{1}{n}\sum_{i=1}^{n}\log\frac{1}{1+\sum_{j=1,j\neq i}^{n} e^{W_i^T W_j-\ell^2}} \\ &= \frac{1}{n}\sum_{i=1}^{n}\log\left(1+\sum_{j=1,j\neq i}^{n} e^{W_i^T W_j-\ell^2}\right).\end{aligned} \tag{3.11}$$

Since $f(x) = e^x$ is a convex function, $\frac{1}{n}\sum_{i=1}^{n} e^{x_i} \geq e^{\frac{1}{n}\sum_{i=1}^{n} x_i}$, then there is,

$$\mathcal{L}_{\mathcal{S}} \geq \frac{1}{n}\sum_{i=1}^{n}\log\left(1+(n-1)e^{\frac{1}{n-1}\sum_{j=1,j\neq i}^{n}(W_i^T W_j-\ell^2)}\right). \tag{3.12}$$

The equality holds if and only if all $W_i^T W_j, 1 \leq i < j \leq n$ have the same value, i.e., features from different classes have the same distance. Unfortunately, in d-dimension space, there are only $d+1$ unique vertices to ensure that every two vertices have the same distance. All these vertices will form a regular d-simplex [134], e.g., a regular 2-simplex is an equilateral triangle and a regular 3-simplex is a regular tetrahedron. Since the class number is usually much bigger than the dimension of the feature in face verification datasets, this equality actually cannot hold in practice. One improvement over this inequality is taking the feature dimension into consideration because the feature dimension term in this step has been omitted.

Similar with $f(x) = e^x$, the softplus function $s(x) = \log(1 + Ce^x)$ is also a convex function when $C > 0$, so that $\frac{1}{n}\sum_{i=1}^{n}\log(1 + Ce^{x_i}) \geq \log(1 + Ce^{\frac{1}{n}\sum_{i=1}^{n} x_i})$, then there is,

$$\begin{aligned}\mathcal{L}_{\mathcal{S}} &\geq \log\left(1 + (n-1)\, e^{\frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}(W_i^T W_j - \ell^2)}\right) \\ &= \log\left(1 + (n-1)\, e^{\left(\frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n} W_i^T W_j\right) - \ell^2}\right).\end{aligned} \tag{3.13}$$

This equality holds if and only if $\forall W_i$, the sums of distances to other class' weight $\sum_{j=1,j\neq i}^{n} W_i^T W_j$ are all the same.

Note that

$$\|\sum_{i=1}^{n} W_i\|_2^2 = n\ell^2 + \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n} W_i^T W_j, \tag{3.14}$$

so

$$\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n} W_i^T W_j \geq -n\ell^2. \tag{3.15}$$

The equality holds if and only if $\sum_{i=1}^{n} W_i = \mathbf{0}$. Thus,

$$\begin{aligned}\mathcal{L}_{\mathcal{S}} &\geq \log\left(1 + (n-1)\, e^{-\frac{n\ell^2}{n(n-1)} - \ell^2}\right) \\ &= \log\left(1 + (n-1)\, e^{-\frac{n}{n-1}\ell^2}\right).\end{aligned} \tag{3.16}$$

Even though reading the proof need patience, it is still encouraged that readers should read it because they may get a better understanding of the hypersphere manifold from it.

This bound implies that if just normalizing the features and weights to 1, the Softmax loss will be trapped at a very high value on the training set, even if no

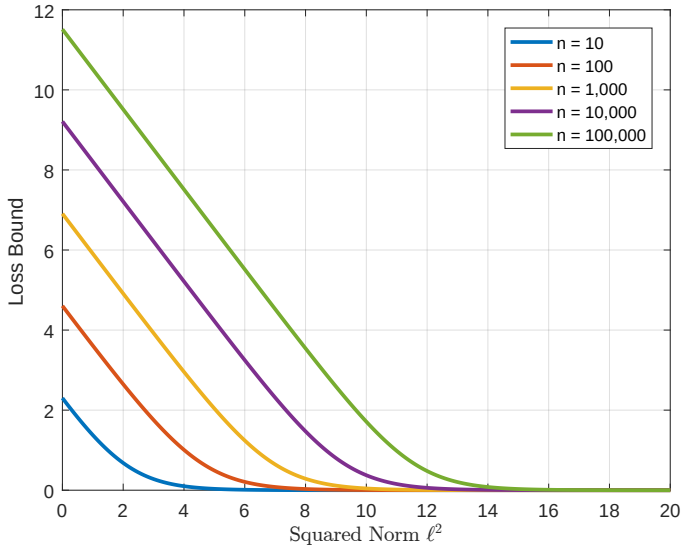


Figure 3.13: The softmax loss' lower bound as a function of features and weights' norm. Note that the x-axis is the squared norm ℓ^2 because the scale parameter is added directly on the cosine distance in practice.

regularization is applied. For a real example, if training the model on the CASIA-Webface dataset ($n = 10575$), the loss will decrease from about 9.27 to about 8.50. The bound for this condition is 8.27, which is very close to the real value. This suggests that our bound is very tight. To give an intuition for the bound, Fig 3.13 also plots the curve of the bound as a function of the norm ℓ.

After the bound is obtained, the solution to the convergence problem is clear. By normalizing the features and columns of weight to a bigger value ℓ instead of 1, the softmax loss can continue to decrease. In practice, this may be implemented by directly appending a scale layer after the cosine layer. The scale layer has only one learnable parameter $s = \ell^2$. It may also be fixed to a value that is large enough referring to Figure 3.13, say 20 or 30 for a different class number. However, it is

preferred to make the parameter automatically learned by back-propagation instead of introducing a new hyper-parameter. Finally, the Softmax loss with cosine distance is defined as

$$\mathcal{L}_{\mathcal{S}\prime} = -\frac{1}{m}\sum_{i=1}^{m} log\frac{e^{s\tilde{W}_{y_i}^T\tilde{\mathbf{f}}_i}}{\sum_{j=1}^{n} e^{s\tilde{W}_j^T\tilde{\mathbf{f}}_i}}, \tag{3.17}$$

where $\tilde{\mathbf{x}}$ is the normalized $\mathbf{x}$. From now on, a deep face network is trained with our normalization layer. Once it is trained, it will be fine-tuned into other domain of facial recognition.

3.2.3 From Softmax loss to regression loss

Similar to conventional regression models, a regression net minimizes the Mean Square Error (MSE) loss defined as

$$\mathcal{L}_{R_{MSE}} = \frac{1}{N}(\sigma(\mathbf{w}^T\mathbf{x}) - \tilde{y})^2 \tag{3.18}$$

where $\mathbf{x}$ is the output vector of the hidden FC layer, $\mathbf{w}$ is a vector of real-valued weights, $\tilde{y}$ is the ground-truth label, and $\sigma(\cdot)$ is a sigmoid activation function $\sigma(x) = \frac{5}{1+e^{-x}}$. $\sigma(\cdot)$ is used to truncate the output of the second FC layer to be in the range of pain intensity $[0, 5]$. Here the bias term is omitted for elegance. The gradient exploding problem often happens due to the relatively large gradient magnitude during initial iterations. This phenomenon is also described in [135]. To solve this problem, following [135], the proposed model applied the smooth $\ell 1$ loss which makes the gra-

dient smaller than the case with the MSE loss when the absolute error $|\sigma(\mathbf{w}^T\mathbf{x}) - \tilde{y}|$ is large. Different from [135], our regressor outputs a scalar instead of a vector. It is a compromise between squared and absolute error losses:

$$\mathcal{L}_R = \begin{cases} 0.5|\sigma(\mathbf{w}^T\mathbf{x}) - \tilde{y}|^2, & if \quad |\sigma(\mathbf{w}^T\mathbf{x}) - \tilde{y}| < t \\ |\sigma(\mathbf{w}^T\mathbf{x}) - \tilde{y}| - t + 0.5t^2, & otherwise \end{cases} \tag{3.19}$$

where t is the turning point of the absolute error between the squared error function and the absolute error function. It has a flavor with the Huber loss. When $t = 1$, it works similarly with MSE loss since the error is usually below 1. When $t = 0$, it is equivalent to the Mean Absolute Error (MAE) loss.

3.2.4 Regularization using the Center loss

Since the pain intensity is labeled as discrete values in the Shoulder-Pain dataset, it is natural to regularize the network to make the regressed values to be 'discrete' - during training, to make same-intensity's regressed values compact (see Fig. 3.14).

The model uses the center loss [105] which minimizes the within-class distance as

$$\mathcal{L}_C = \|\mathbf{x} - \mathbf{c}_{\tilde{y}}\|_p^p, \tag{3.20}$$

where $c_{\tilde{y}}$ represents the center for class $\tilde{y}$ and is essentially the mean of features per class. p denotes the norm and is typically $\ell 1$ or $\ell 2$. It is observed from experiments that the center loss shrinks the distances of features that have the same label, which is illustrated in Fig. 3.14. To relate it to the literature, it is a similar idea to the

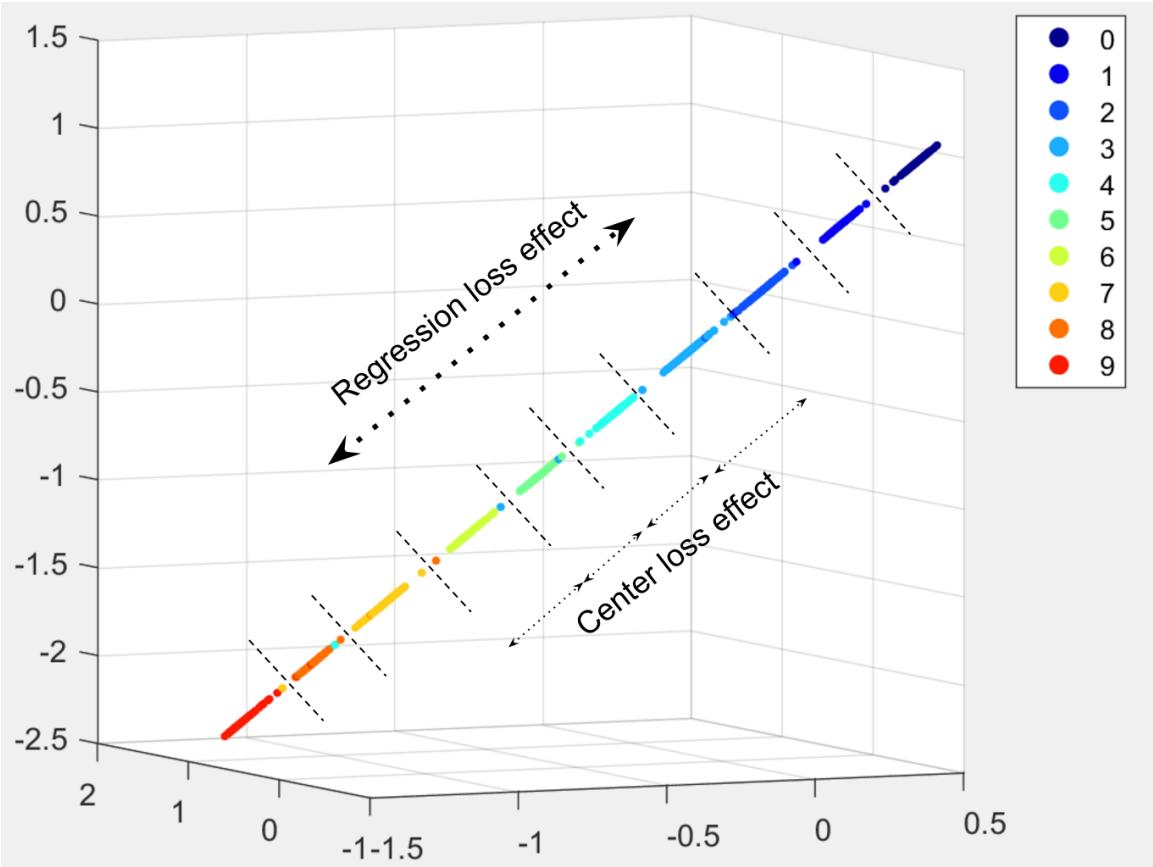


Figure 3.14: Illustration of how the regularized loss functions works. Each point represents a feature vector projected into the 3D space and thus is called feature point. By a regression loss, a linear projection is found to project the feature vectors to one-dimension values. The calibration of the coordinate axis is not uniform because the Sigmoid activation is used, which is not a linear function. While the regression loss spreads the feature points out all over the axis, the center loss induces feature points associated with close values to be distributed as adjacent as possible.

Linear Discriminant Analysis yet without minimizing between-class distances. It also has a flavor of the k-means clustering yet in a supervised way.

Now, the center loss is added to the regression loss after the hidden FC layer to induce the loss $\mathcal{L} = \mathcal{L}_R + \lambda\mathcal{L}_C$ where λ is a coefficient. Thus, the supervision of the regularizer is applied to the features. Different from [105], the model jointly learns

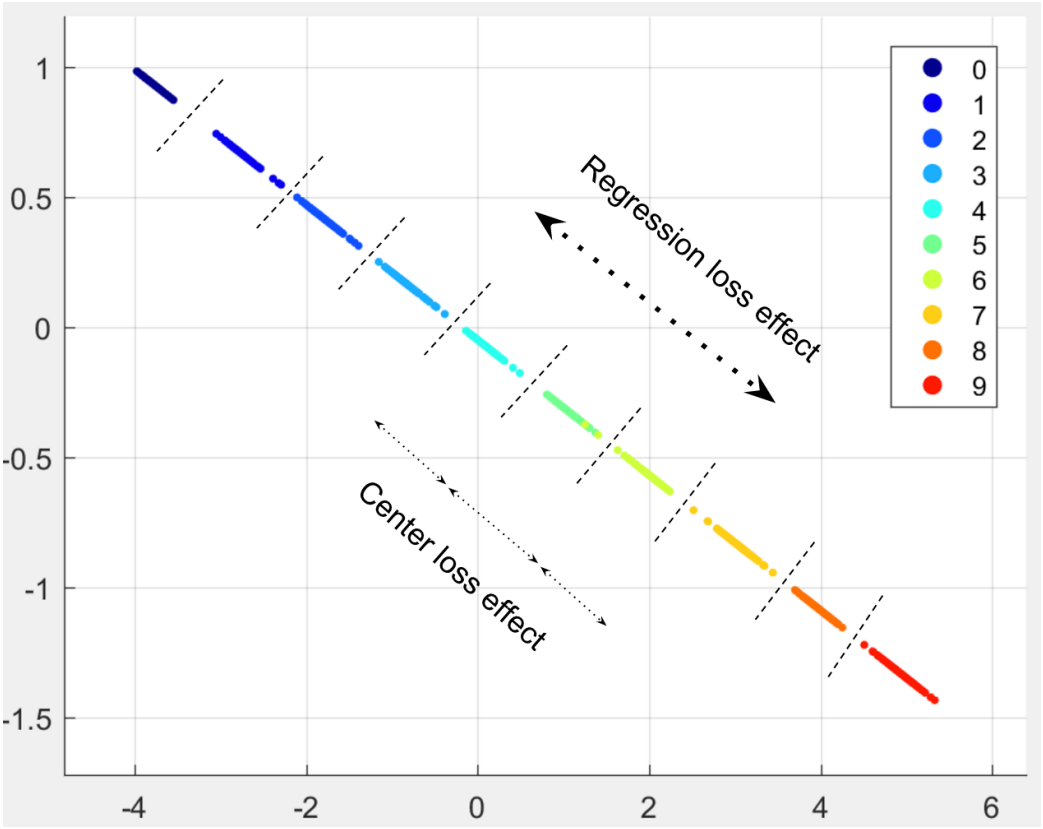


Figure 3.15: Illustration of the regularized loss with features projected to 2D space. the centers and minimizes within-class distances by gradient descent, while [105]'s centers are learned by moving average.

3.2.5 Experiments

Our network is tested on the Shoulder-Pain dataset [136] that contains 200 videos of 25 subjects and is widely used for benchmarking the pain intensity estimation. The dataset comes with four types of labels. The three datasets that are annotated online during the video collection are the sensory scale, affective scale and visual analog

scale ranging from 0 (*i.e.*, no pain) to 15 (*i.e.*, severe pain). In addition, observers rated pain intensity offline from recorded videos ranging from 0 (no pain) to 5 (severe pain). In the same way with previous works [106, 107, 137], the proposed model takes the same online label and quantifies the original pain intensity in the range of $[0, 15]$ to be in the range $[0, 5]$.

3.2.5.1 Implementation

The face verification network [105] is trained on CASIA-WebFace dataset [138], which contains 494,414 training images from 10,575 identities. To be consistent with face verification, the same pre-processing is performed on the images of Shoulder-Pain dataset. To be specific, the Multi-Task CNN (MTCNN) model [139] is leveraged to detect faces and facial landmarks. Then the faces are aligned according to the detected landmarks.

The learning rate is set to 0.0001 to avoid huge modification on the convolution layers. The network is trained over 5,000 iterations, which is reasonable for the networks to converge observed in a few cross-validation folds. The center loss coefficient λ is set to be 0.01 according to the practice.

3.2.5.2 Weighted evaluation metrics

Labels in the Shoulder-Pain dataset are highly imbalanced, as 91.35% of the frames are labeled as pain intensity 0. Thus, it is relatively safe to predict the pain intensity

to be zero.

To fairly evaluate the performance, the weighted version of evaluation metrics is proposed, *i.e.*, weighted MAE (wMAE) and weighted MSE (wMSE) to address the dataset imbalance issue. For example, the wMAE is simply the mean of MAE on each pain intensity. In this way, MAE is weighted by the population of each intensity. We also use the Pearson correlation coefficient (PCC) which measures the curve trend similarity.

Two techniques are applied to sample the training data to make our training set more consistent with the new metrics. First, the redundant frames are eliminated on the sequences following [106]. If the intensity remains the same for more than 5 consecutive frames, the first one is chosen as the representative frame. Second, during training, images from the 6 classes are uniformly sampled to feed into the network. In this way, what the neural network 'see' is a totally balanced dataset.

3.2.5.3 Evaluation using unweighted metrics

Cross validation is a conventional way to address over-fitting small dataset. In our case, 5-fold cross validations are performed 25 times on the Shoulder-Pain dataset which contains 25 subjects. This setting is exactly the leave-one-subject-out setting in OSVR [106] except that OSVR's experiments exclude one subject whose expressions do not have noticeable pain (namely 24-fold). Each time, the videos of one subject are reserved for testing. All the other videos are used to train the deep regression

Methods	MAE↓	MSE↓	PCC↑
smooth $\ell 1$	0.416	1.060	0.524
$\ell 1$+ $\ell 1$ center loss	***0.389***	0.820	0.603
smooth $\ell 1$ + $\ell 1$ center loss	0.456	***0.804***	***0.651***
smooth $\ell 1$ + $\ell 2$ center loss	0.435	0.816	0.625
OSVR-$\ell 1$ ([106] CVPR'16)	1.025	N/A	0.600
OSVR-$\ell 2$ ([106] CVPR'16)	0.810	N/A	0.601
RCR ([107] CVPR'16w)	N/A	1.54	***0.65***
All Zeros (trivial solution)	0.438	1.353	N/A

Table 3.1: Performance of our regression network and related works on the Shoulder-Pain dataset for the estimation of pain intensity (*i.e.*, pain expression intensity). MAE is short for mean absolute error deviated from the ground-truth labels over all frames per video. MSE is mean squared error which measures the curve fitting degree. PCC is Pearson correlation coefficient which measures the curve trend similarity (↑ indicates the larger, the better). The best is highlighted in bold.

network. The performance is summarized in Table 3.1. It can be concluded that our algorithm performs best or equally best on various evaluation metrics, especially the combination of the smooth $\ell 1$ loss and $\ell 1$ center loss. Note that OSVR [106] uses hand-crafted features concatenated from landmark points, Gabor wavelet coefficients and LBP + PCA.

3.2.5.4 Evaluation using weighted metrics

In the following, Table 3.1 provides the performance of predicting all zeros as a baseline. Interestingly, on the metrics MAE and MSE, zero prediction performs much better than several state-of-the-art algorithms. Now, using the new proposed metrics, the performance is summarized in Table 3.2. The performance of previous work OSVR [106] is no longer below that of predicting all zeros. It can also be seen from Table 3.2 that the uniform class sampling strategy does help a lot on the new evaluation metrics. Moreover, the evaluation program has been provided in our project page and encourage future works to report their performance with the new evaluation metrics.

3.2.6 Summary

The problem that motivates the work of this subsection is that expertise is needed to label the pain. Given the restriction of labeled data which prevents us from directly training a deep pain level regressor, fine-tuning from a data-extensive pre-trained domain such as face verification can alleviate the problem. In this section, a face verification network is regularized for the pain level regression. In particular, the Smooth $\ell 1$ Loss is introduced to (continuous-valued) pain level regression. The center loss is also introduced as a regularizer to induce concentration on discrete values. The fine-tuned regularized network with a regression layer is tested on the Shoulder-Pain

Methods	wMAE↓	wMSE↓
smooth $\ell 1$	1.596	4.396
$\ell 1$ + $\ell 1$ center loss	1.388	3.438
smooth $\ell 1$ + $\ell 1$ center loss	1.289	2.880
smooth $\ell 1$ + $\ell 2$ center loss	1.324	3.075
$\ell 1$ + $\ell 1$ cente loss + sampling	1.039	1.999
smooth $\ell 1$ + $\ell 1$ center loss + sampling	***0.991***	***1.720***
OSVR-$\ell 1$ ([106] CVPR'16)	1.309	2.758
OSVR-$\ell 2$ ([106] CVPR'16)	1.299	2.719
All Zeros (trivial solution)	2.143	7.387

Table 3.2: Performance of our network when evaluated using the weighted MAE and weighted MSE. 'sampling' means the uniform class sampling technique is applied. Notably, $\ell 1$ center loss and sampling incrementally boost the performance.

dataset and achieves state-of-the-art performance on pain level estimation.

On the other hand, unsupervised learning does not rely on training data. Indeed, discrete-valued regression is a good test bed for center-based clustering. Although regularizing a supervised deep network is intuitive, its performance is rather empirical. In the future, insights are needed about when and why it may function as transfer learning. Note that no temporal information is modeled. As pain is temporal and subjective, prior knowledge about the stimulus needs to be incorporated to help quantify individual differences.

3.3 Conclusion

The first example work essentially is video summarization in terms of keyframe selection. Pose has been used to select keyframes. Two challenges are addressed for measuring the similarity of the subject identities in practical video-based face recognition - the variation of the head pose in uncontrolled environments and the computational expense of processing videos. Since the frame-wise feature mean is unable to characterize the pose diversity among frames, the model defines and preserves the overall pose diversity and closeness in a video. Then, identity will be the only source of variation across videos since the pose varies even within a single video. Instead of simply using all the frames, the model selects those faces whose pose point is closest to the centroid of the K-means cluster containing that pose point. Then, the model represents a video as a bag of frame-wise deep face features while the number of features has been reduced from hundreds to K. Since the video representation can well represent the identity, now the model measures the subject similarity between two videos as the max correlation among all possible pairs in the two bags of features. On the official 5,000 video-pairs of the YouTube Face dataset for face verification, our algorithm achieves a comparable performance with VGG-face that averages over deep features of all frames. Other vision tasks can also benefit from the generic idea of employing geometric cues to improve the descriptiveness of deep features. This work has been published in our paper [140].

There exist other criteria for the subset selection. Keyframe selection has been

studied a lot in video-based face recognition, such as frame weighting [141, 142], face clustering on Euclidean distances [143], clustering based on geodesic distances on manifolds (LLE [144], Isomap [145]), selection based on difference in face space and image quality [146], selection based on pose and motion blur [147]. In particular, a strategy of the same kind with ours is based on the distance between the class means [141].

The cosine similarity or correlation both are well-defined metrics for measuring the similarity of two images. A simple adaptation to videos will be randomly sampling a frame from each of the video. However, the correlation between two random image samples might characterize cues other than identity (say, the pose similarity). There are existing works on measuring the similarity of two videos using manifold-to-manifold distance [101]. However, the straightforward extension of image-based correlation is preferred for its simplicity, such as temporal max or mean pooling [148]. The impact of different spatial pooling methods in CNN such as mean pooling, max pooling and L-2 pooling, has been discussed in the literature [149, 150]. However, pooling over the time domain is not as straightforward as spatial pooling. The frame-wise feature mean is a straightforward video-level representation and yet not a robust statistic. Despite that, temporal mean pooling is conventional to represent a video such as average pooling for video-level representation [151], mean encoding for face recognition [152], feature averaging for action recognition [153] and mean pooling for video captioning [154].

Motivating from the similarity measure, the ℓ_2 Softmax loss [127] and our normalized layer [104] have been recently developed. The normalized layer has been used in the second example work in this chapter. In a typical face verification method, feature normalization is a critical step for boosting performance. This motivates us to introduce and study the effect of normalization during training. But it is found that this is non-trivial, despite normalization being differentiable. Four issues are identified related to normalization through mathematical analysis, which yields understanding and helps with parameter settings. Based on this analysis two strategies are proposed for training using normalized features. The first is a modification of the Softmax loss, which optimizes cosine similarity instead of inner-product. This work has been published in our paper [104].

As regards regularizing deep networks, there exists recent work that regularizes deep face recognition nets for expression classification - FaceNet2ExpNet [103]. During pre-training, they train convolutional layers of the expression net, regularized by the deep face recognition net. In the refining stage, they append fully-connected (FC) layers to the pre-trained convolutional layers and train the whole network jointly. This work has been published in our paper [155].

Chapter 4

Image-set Representation of Face Videos via Inter-frame Models

Similar with the regularizing deep network work in the previous chapter, here the same challenge is faced to recognize facial expression: limited annotated data available for the recognition of facial expression and particularly action units makes it hard to train a deep network which can learn disentangled invariant features. However, a supervised linear model is undemanding in terms of training data. In this chapter, an elegant linear model is proposed to untangle facial actions from expressive face videos which contain a mixture of linearly representable attributes. Previous attempts require an explicit decoupling of identity and expression which is practically inexact. Instead, the low-rank property is exploited across frames to implicitly subtract the intrinsic neutral face, which is modeled jointly with sparse representation only on the

residual expression components. On CK+, our one-shot C-HiSLR on raw-face pixel intensities performs far more competitive than conventional shape+SVM models with landmark detection and two-stepped SRC of the same type yet applied on manually prepared expression components. It is also comparable with the piecewise linear model DCS and temporal models such as CRF and Bayes nets. It is applied to Action Unit (AU) recognition on MPI-VDB achieving a decent performance. As expression is a mixture of AUs, the result gives hopes of approximating a nonlinear expression using a piecewise linear model.

Furthermore, a video is a recursively measured signal where frames are highly correlated with structured sparsity and low-rankness. A simple example is the facial expression - multiple measurements of a face. Several salient facial action units (AU) are often enough for a correct expression recognition. It is hoped that AUs are not stored when the face remains neutral until they become salient when expression occurs, as well as that the recognizer is still able to restore historic salient AUs. A temporal memory mechanism is appealing for a real-time system to reduce rich redundancy in information coding. The expression recognition problem is formulated as a video Sparse Representation based Classification (SRC) with Long Short-Term Memory (LSTM), which is applicable for human actions. Illustrating examples are taken from the MPI Face Video Database (MPI-VDB). The second section compares the proposed sparse coding with temporal modeling using LSTM against the baseline of sparse coding with simultaneous recursive matching pursuit (SRMP).

4.1 Collaborative principal recovery and sparsely coding residual action

Following the previous section, this section takes a close look at the facial expression videos. Instead of treating video frames independently, the frames are highly correlated so that an underlying low-rank matrix exists if each frame is represented as a vector and the vectors are arranged to form a matrix. Low-rank matrix approximation does not necessarily give us a visually meaningful object so once again other prior knowledge is applied such as structured sparsity in the representation learning.

4.1.1 Problem and background

In this section, the problem is to recognize facial actions given a face video and action categories in the granularity of either the holistic expression (Fig. 4.1) or Action Units (AU, muscles, Fig. 4.2). Paul Ekman defines 6 basic expressions including *surprise, sadness, disgust, anger, fear* and *happiness* and Facial Action Coding System (FACS) using which almost any expression can be coded. A face video is a perspective projection over time to a 2-D plane for a face varying with AU surfaces. Suppose that it is generated by a weighted linear mixture of attributes including AUs, identity and viewpoint, it is unrealistic to expect raw faces with the same expression to lie on a subspace as assumed in Eigenface [91] and sparse representation [73].

However, Fig.4.1 implies that a face can be separated into a Principal Component

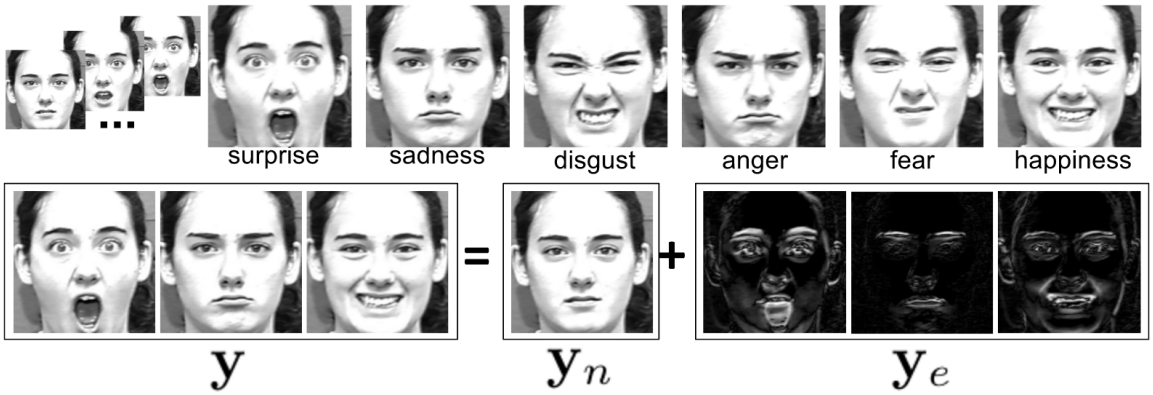


Figure 4.1: Neutral face $\mathbf{y}_n$ encodes identity; residue $\mathbf{y}_e$ encodes expression. $\mathbf{y}_e$ is linearly representable while the raw $\mathbf{y}$ is not.

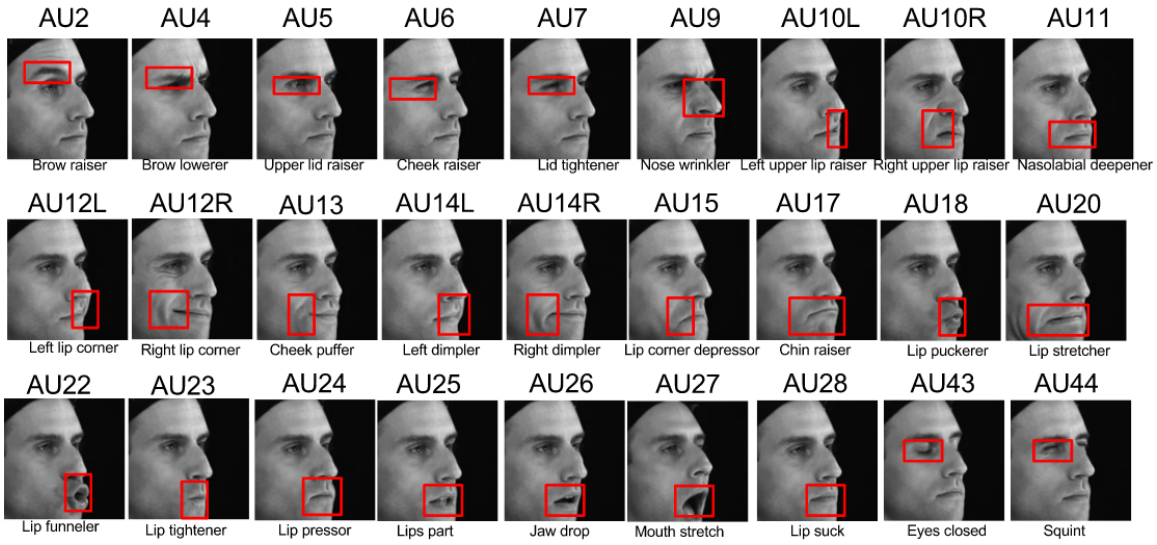


Figure 4.2: Action unit number and FACS name shown using images from MPI-VDB with 27 distinct AUs. The peak frame is shown. AU12L and AU12R are distinct; similar for AU14.

of the neutral face encoding identity cues and a residual expression component encoding motion cues such as the brow, cheek and lip relating to AUs. For identification, PCs across well-normalized images of different people are known as Eigenfaces. Similarly, PCs across frames of the same person are Eigenfaces for that specific person at the different time. Thus, the neutral face corresponding with the first PC is supposed to encode identity cues. For that reason, an expressive face in identification is warped into a neutral face using a piece-wise linear transformation such as AAM.

Without decoupling attributes, a similar identity can confuse a similar expression. Once getting rid of the intrinsic neutral face, the problem is simplified to recognizing the expression component. Given videos with only identity and expression varying, two-piece linear representation is feasible, which follows the idea of piecewise linear representation based on Independent/Morphological Component Analysis verified for dual purposes [156]. As coupled representations are not needed, the neutral face is separated as an interference and stack the neutral face vector across frames as a low-rank matrix, ideally with rank 1. While theoretically the low-rank matrix recovery (robust PCA) is exact under conditions, the Principal Component Pursuit algorithm [77] is of approximate nature due to relaxation and alternative optimization. Avoiding such an explicit PCP is beneficial in precision and computation. **Both the separation and linear representation are performed in a joint model named SLR** [100]. Since all frames collaboratively yet redundantly represent an expression, in SLR frames are sampled to represent an expression component as a **joint-sparse**

coefficient matrix [75] **which induce consistent classifications across frames**. The class-level sparsity is 1 and the coefficient matrix exhibits group sparsity, because ideally non-zero coefficients all drop to the ground-truth class. But coefficient vectors share class-wise yet not necessarily atom-wise sparsity [78]. **In C-HiSLR** [100], **The group sparsity and atom-wise sparsity are both enforced.** The next section explains how to model $\mathbf{X}$ using $\mathbf{Y}$ and training data $\mathbf{D}$, which contains $K \in \mathbb{Z}^+$ types of **actions**. The task is to classify a test video as one of the K classes.

4.1.2 Sparse representation related works

Linear representation [73,156,157] models are mostly appearance-based (even raw signals). In computer vision, it is standard to represent shape using key points [158–160]. For facial actions, videos are treated as multichannel signals [75], different from image-based methods [156, 157]. Furthermore, [110, 160, 161] models temporal dynamics such as shape evolving [161]. One sort of nonlinear models is kernel-based methods [160] while another is deep learning [116,118,162]. See details in Sec. 4.1.6.1. Particularly for videos, Aswin Sankaranarayanan *et. al.* develop a new framework for video compressed sensing for dynamic textured scenes that models the evolution of the scene as a linear dynamical system [163].

Piecewise linear unmixing. [156] first separates the neural face explicitly and then discriminates expression components.

Nonlinear unmixing. [164] jointly selects and disentangles expression/AU specific features via kernel SVM. Deep learning [116, 165] using auto-encoders or adversarial

training has shown the capability to disentangle facial *expression* from *identity* and *pose*. But limited labeled training data are available for facial expressions and AUs, unlike identification.

Linear signal representation revisited. In a supervised linear model for classification tasks, observing a random signal $\mathbf{y}$, it is hoped to send the classifier a discriminative compact representation $\mathbf{x}$ over a dictionary $\mathbf{D}$ formed by training samples of all classes such that $\mathbf{Dx} = \mathbf{y}$ where $\mathbf{x}$ is computed by pursuing the best reconstruction. If $\mathbf{D}$ is under-complete, then a closed-form approximate solution can be obtained by Least-Squares. Normally it is easy to obtain more training data than the dimensionality so $\mathbf{D}$ is over-complete. Then a Tikhonov regularizer is added or a sparse usage of $\mathbf{D}$ is pursued. Sparse Representation based Classification [73] (SRC) expresses a test sample $\mathbf{y}$ as a linear combination $\mathbf{y} = \mathbf{Dx}$ of training samples stacked column-wise in a dictionary $\mathbf{D}$. Presumably, non-zero weight coefficients drop to the ground-truth class, which induces a sparse coefficient vector or the so-called sparse representation. In practice, non-zero coefficients also drop to other classes due to noises and correlations among classes. For classification, SRC evaluates which class leads to the minimum reconstruction error which works as a max-margin-like classifier.

4.1.3 SLR: Sparse representation and Low-Rankness

First, an explicit representation $\mathbf{Y}$ of an expressive face is needed. A matrix $\mathbf{Y} \in \mathbb{R}^{d \times \tau}$ is an arrangement of d-dimensional feature vectors $\mathbf{y} \in \mathbb{R}^d$ $(i = 1, 2, ..., \tau)$

of τ frames: $\mathbf{Y} = \left[\mathbf{Y}_1|...|\mathbf{Y}_\tau\right]_{d\times\tau}$. The proposed model's discriminative power is shown by using pixel intensities and thus gray-scale images are shown in figures. A compact latent representation $\mathbf{X} \in \mathbb{R}^{n\times\tau}$ of a test expression component $\mathbf{Y}_e \in \mathbb{R}^{d\times\tau}$ is pursued as a linear combination of expressions in a dictionary $\mathbf{D} \in \mathbb{R}^{d\times n}$:

$$\mathbf{Y}_e = \mathbf{D}\mathbf{X}.$$

Since an expressive face $\mathbf{y} = \mathbf{y}_e + \mathbf{y}_n$ is a superposition of an expression $\mathbf{y}_e \in \mathbb{R}^d$ and a neutral face $\mathbf{y}_n \in \mathbb{R}^d$, there is

$$\mathbf{Y} = \mathbf{Y}_e + \mathbf{L},$$

where $\mathbf{L} \in \mathbb{R}^{d\times\tau}$ is ideally τ-times repetition of the column vector of a neutral face $\mathbf{y}_n \in \mathbb{R}^d$. Presumably $\mathbf{L} = \left[\mathbf{y}_n|...|\mathbf{y}_n\right]_{d\times\tau}$. As shown in Fig. 4.3, $\mathbf{X}$ subjects to

$$\mathbf{Y} = \mathbf{D}\mathbf{X} + \mathbf{L}, \tag{4.1}$$

where the dictionary matrix $\mathbf{D}_{d\times n}$ is an arrangement of all sub-matrices $\mathbf{D}_{[j]}$, $j = 1, ..., \lfloor\frac{n}{\tau}\rfloor$. *Only for training*, there is $\lfloor\frac{n}{\tau}\rfloor$ training expressions with neutral faces subtracted. The above constraint of $\mathbf{X}$ characterizes an affine transformation from the latent representation $\mathbf{X}$ to the observation $\mathbf{Y}$.

Let $\mathbf{X}$ and $\mathbf{Y}$ be written in the homogeneous form, there is

$$\begin{bmatrix} \mathbf{Y}_{d\times\tau} \\ \mathbf{1}_{1\times\tau} \end{bmatrix} = \begin{bmatrix} \mathbf{D}_{d\times n} & \left(\mathbf{y}_n\right)_{d\times 1} \\ \mathbf{0}_{1\times n} & 1 \end{bmatrix} \times \begin{bmatrix} \mathbf{X}_{n\times\tau} \\ \mathbf{1}_{1\times\tau} \end{bmatrix}. \tag{4.2}$$

In the ideal case with $rank(\mathbf{L}) = 1$, if the neutral face $\mathbf{y}_n$ is pre-obtained [156, 157], it is trivial to solve for $\mathbf{X}$. Normally, $\mathbf{y}_n$ is unknown and $\mathbf{L}$ is not with rank 1 due to noises. As $\mathbf{X}$ is supposed to be sparse and $rank(\mathbf{L})$ is expected to be as small as

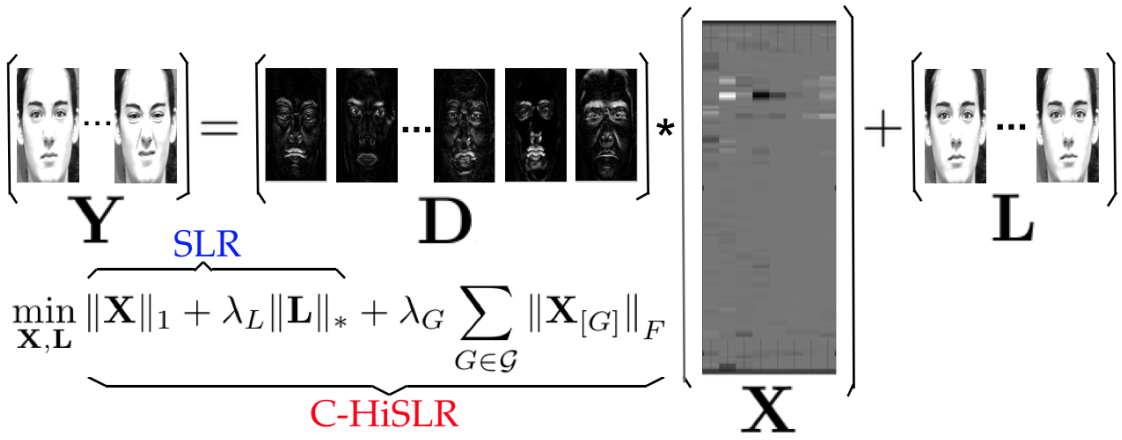


Figure 4.3: Pictorial illustration of the linear constraint of the proposed model shown for *disgust*. $\mathbf{D}$ is prepared and simply fixed. Depending on the objective the model has two versions.

possible (maybe even 1), intuitively our objective is to

$$\min_{\mathbf{X},\mathbf{L}} sparsity(\mathbf{X}) + \lambda_L \cdot rank(\mathbf{L}),$$

where $rank(\mathbf{L})$ can be seen as the sparsity of the vector formed by the singular values of $\mathbf{L}$. Here λ_L is a non-negative weighting parameter to be tuned. When $\lambda_L = 0$, the optimization problem reduces to that in SRC. With both terms relaxed to be convex norms, now it is alternatively solving

$$\min_{\mathbf{X},\mathbf{L}} \|\mathbf{X}\|_1 + \lambda_L \|\mathbf{L}\|_* \tag{4.3}$$

where $\|\cdot\|_1$ is the entry-wise ℓ_1 matrix norm, whereas $\|\cdot\|_*$ is the Schatten ℓ_1 matrix norm (nuclear norm, trace norm) which can be seen as applying ℓ_1 norm to the vector of singular values. Now, the proposed joint SLR model is expressed as

$$\min_{\mathbf{X},\mathbf{L}} \|\mathbf{X}\|_1 + \lambda_L \|\mathbf{L}\|_* \quad s.t. \quad \mathbf{Y} = \mathbf{D}\mathbf{X} + \mathbf{L} \tag{4.4}$$

Now the model solves (4.4) for matrices $\mathbf{X}$ and $\mathbf{L}$ by the Alternating Direction Method of Multipliers (ADMM) (see Sec. 4.1.5).

4.1.4 C-HiSLR: Collaborative-Hierarchical SLR

If there is no a low-rank term $\mathbf{L}$, Eqn. (4.4) becomes a problem of multi-channel LASSO. For single channels, Group LASSO has explored the group structure and yet does not enforce sparsity within a group, while Sparse Group Lasso yields an atom-wise sparsity as well as a group sparsity. Then, [78] extends Sparse Group LASSO to multichannel, resulting in a Collaborative-Hierarchical LASSO (C-HiLasso) model. For our problem, the model does need $\mathbf{L}$, which induces a Collaborative-Hierarchical Sparse and Low-Rank (C-HiSLR) model:

$$\begin{aligned} &\min_{\mathbf{X},\mathbf{L}} \|\mathbf{X}\|_1 + \lambda_L \|\mathbf{L}\|_* + \lambda_G \sum_{G\in\mathcal{G}} \|\mathbf{X}_{[G]}\|_F \\ &s.t. \quad \mathbf{Y} = \mathbf{D}\mathbf{X} + \mathbf{L} \end{aligned} \tag{4.5}$$

where $\mathbf{X}_{[G]}$ is the sub-matrix formed by all the rows indexed by elements in group $G \subseteq \{1, ...,\mathrm{n}\}$. For a group G of indices, the sub-dictionary of columns indexed by G is denoted as $\mathbf{D}_{[G]}$. Given a non-overlapping partition $\mathcal{G} = \{G_1, \cdots, G_K\}$ of $\{1, ..., n\}$, $\mathbf{D}\mathbf{X} = \mathbf{D}_{[G_1]}\mathbf{X}_{[G_1]} + ... + \mathbf{D}_{[G_K]}\mathbf{X}_{[G_K]}$ is a group-wise linear representation. The Frobenius norm $\|\cdot\|_F$ is the Schatten ℓ_2 matrix norm and characterizes a matrix's energy (entry-wise ℓ_2 norm). λ_G is a non-negative weighting parameter for the group regularizer, which is generalized from an ℓ_1 regularizer (consider $\mathcal{G} = \big\{\{1\}, \{2\}, ..., \{n\}\big\}$

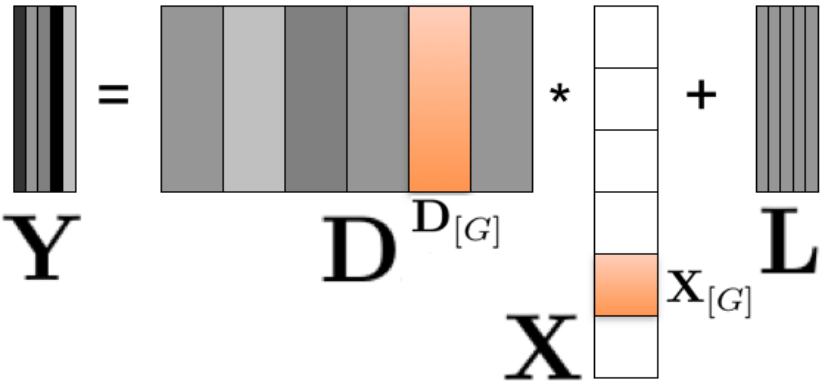


Figure 4.4: Pictorial illustration of the group sparsity in C-HiSLR.

for singleton groups) [78]. Note that C-HiSLR degenerates into sparse group LASSO when $\lambda_L = 0$ and SLR when $\lambda_G = 0$.

4.1.5 Classification and optimization

Following SRC, for each class $c \in \{1, 2, ..., K\}$, let $\mathbf{D}_{[G_c]}$ denote the sub-matrix of $\mathbf{D}$ which consists of all the columns of $\mathbf{D}$ that correspond to expression class c and similarly for $\mathbf{X}^{[G_c]}$. $\mathbf{Y}$ is classified by assigning it to the class with minimal residual as $c^* = \arg\min_c r_c(\mathbf{Y}) := ||\mathbf{Y} - \mathbf{D}_{[G_c]}\mathbf{X}_{[G_c]} - \mathbf{L}||_F$.

Both SLR and C-HiSLR models can be seen as solving

$$\min_{\mathbf{X},\mathbf{L}} f(\mathbf{X}) + \lambda_L \|\mathbf{L}\|_* \quad s.t. \quad \mathbf{Y} = \mathbf{D}\mathbf{X} + \mathbf{L} \tag{4.6}$$

To follow a standard iterative ADMM procedure, the augmented Lagrangian function for (4.6) can be written as

$$\begin{aligned}\mathcal{L}(\mathbf{X},\mathbf{L},\mathbf{\Lambda}) &= f(\mathbf{X}) + \lambda_L||\mathbf{L}||_* \\ &+ \langle \mathbf{\Lambda}, \mathbf{Y}-\mathbf{D}\mathbf{X}-\mathbf{L}\rangle + \frac{\beta}{2}||\mathbf{Y}-\mathbf{D}\mathbf{X}-\mathbf{L}||_F^2,\end{aligned} \tag{4.7}$$

where $\mathbf{\Lambda}$ is the matrix of multipliers, $\langle\cdot,\cdot\rangle$ is the inner product, and β is a positive weighting parameter for the penalty (augmentation). A single update at the k-th iteration includes

$$\mathbf{L}_{k+1} = \arg\min_{\mathbf{L}} \lambda_L||\mathbf{L}||_* + \frac{\beta}{2}||\mathbf{Y}-\mathbf{D}\mathbf{X}_k-\mathbf{L}+\frac{1}{\beta}\mathbf{\Lambda}_k||_F^2 \tag{4.8}$$

$$\mathbf{X}_{k+1} = \arg\min_{\mathbf{X}} f(\mathbf{X}) + \frac{\beta}{2}||\mathbf{Y}-\mathbf{D}\mathbf{X}-\mathbf{L}_{k+1}+\frac{1}{\beta}\mathbf{\Lambda}_k||_F^2 \tag{4.9}$$

$$\mathbf{\Lambda}_{k+1} = \mathbf{\Lambda}_k + \beta(\mathbf{Y}-\mathbf{D}\mathbf{X}_{k+1}-\mathbf{L}_{k+1}). \tag{4.10}$$

The sub-step of solving (4.8) has a closed-form solution:

$$\mathbf{L}_{k+1} = \mathcal{D}_{\frac{\lambda_L}{\beta}}(\mathbf{Y}-\mathbf{D}\mathbf{X}_k+\frac{1}{\beta}\mathbf{\Lambda}_k), \tag{4.11}$$

where $\mathcal{D}$ is the shrinkage thresholding operator. In SLR where $f(\mathbf{X}) = \|\mathbf{X}\|_1$, (4.9) is a LASSO problem, which is solved by using the Illinois fast solver. When $f(\mathbf{X})$ follows (4.5) of C-HiSLR, computing $\mathbf{X}_{k+1}$ needs a Taylor expansion at $\mathbf{X}_k$.

4.1.6 Experiments

In this section, experimental results are presented for the holistic facial expression recognition and the fine-grained facial action unit recognition, respectively.

	An	Co	Di	Fe	Ha	Sa	Su
CHi-SLR:Angry	**0.77**	0.01	0.09	0.02	0	0.07	0.04
Contempt	0.08	**0.84**	0	0	0.03	0.04	0
Disgust	0.05	0	**0.93**	0.01	0.01	0.01	0
Fear	0.09	0.01	0.03	**0.53**	0.12	0.07	0.15
Happy	0.01	0.02	0.01	0.02	**0.93**	0	0.03
Sad	0.19	0.02	0.02	0.05	0	**0.65**	0.07
Surprise	0	0.02	0	0.02	0	0.02	**0.95**
SLR: Angry	**0.51**	0	0.10	0.02	0	0.31	0.06
Contempt	0.03	**0.63**	0.03	0	0.04	0.26	0.01
Disgust	0.04	0	**0.74**	0.02	0.01	0.15	0.04
Fear	0.08	0	0.01	**0.51**	0.03	0.19	0.18
Happy	0	0.01	0	0.03	**0.85**	0.08	0.03
Sad	0.09	0	0.04	0.04	0	**0.70**	0.13
Surprise	0	0.01	0	0.02	0.01	0.02	**0.94**
SRC + neu. [157]: An	**0.71**	0.01	0.07	0.02	0.01	0.03	0.16
Contempt	0.07	**0.60**	0.02	0	0.16	0.03	0.12
Disgust	0.04	0	**0.93**	0.02	0.01	0	0
Fear	0.16	0	0.09	**0.25**	0.25	0	0.26
Happy	0.01	0	0	0.01	**0.96**	0	0.02
Sad	0.22	0	0.13	0.01	0.04	**0.24**	0.35
Surprise	0	0.01	0	0	0.01	0	**0.98**
DTGAN [118]:**0.95**	1.00	0.94	1.00	0.84	1.00	0.89	0.98
3DCNN-DAP:**0.88**	0.91	0.67	0.97	0.80	0.99	0.86	0.96
3DCNN [162]:**0.78**	0.78	0.61	0.97	0.60	0.96	0.57	0.98
H-CRF [160]: **0.88**	0.95	0.82	0.95	0.84	0.95	0.64	0.98
ITBN [110]: **0.86**	0.91	0.78	0.94	0.83	0.90	0.76	0.91
LD-CRF [161]:**0.85**	0.73	0.76	0.81	0.94	0.98	0.77	0.99
AAM [158]: **0.83**	0.75	0.84	0.95	0.65	1.00	0.68	0.96
CLM [159]: **0.74**	0.70	0.52	0.93	0.72	0.94	0.46	0.93
RBF [160]: **0.72**	0.77	0.45	0.82	0.67	0.86	0.62	0.87
DCS [156]: **0.81**	0.76	**N/A**	0.96	0.50	0.98	0.69	0.98
SRC+neu. [157]: **0.67**	0.71	0.60	0.93	0.25	0.96	0.24	0.98
SLR: **0.70**	0.51	0.63	0.74	0.51	0.85	0.70	0.94
C-HiSLR: **0.80**	0.77	0.84	0.93	0.53	0.93	0.65	0.95

Table 4.1: Per-class accuracy on CK+. DCS, SRC, SLR and C-HiSLR uses intensities.

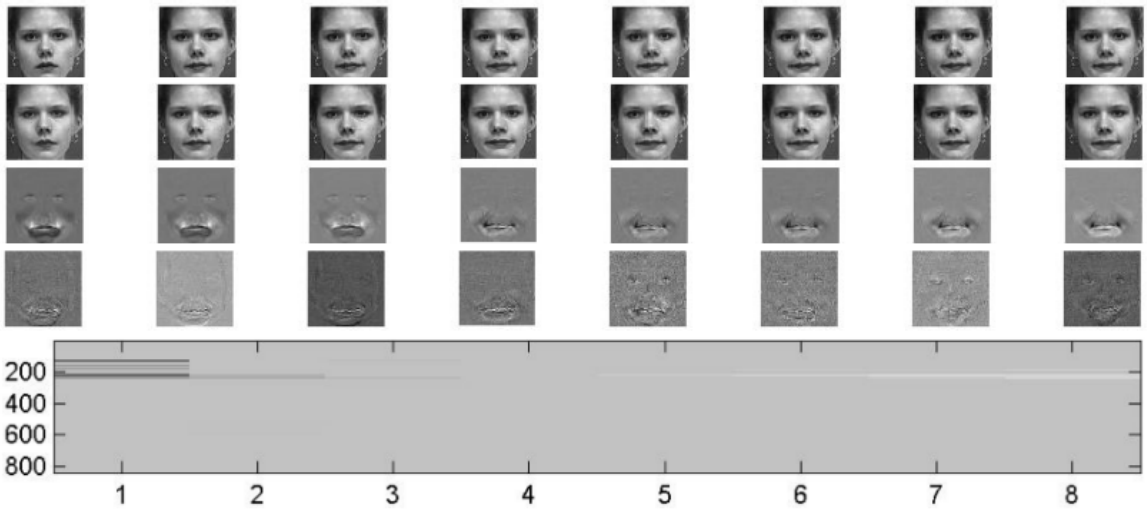


Figure 4.5: Example recovery results of C-HiSLR. First row: the testing sequence. Second row: the recovered $\mathbf{L}$ which is hoped to represent a neutral face. Third row: $\mathbf{AX}$. Fourth row: the residue $\mathbf{Y} - \mathbf{AX} - \mathbf{L}$. Fifth row: the recovered $\mathbf{X}$.

4.1.6.1 Holistic facial expression recognition

Our model is evaluated on expressions (CK+) and action units (MPI-VDB). Images are cropped using the Viola-Jones face detector. Per-class accuracy rates are averaged over 20 runs. Experiments are performed on the CK+ dataset [158] consisting of 321 videos[1]. A test unit contains the last $(\tau_{tst}-1)$ frames together with the first frame, which is **not** explicitly known as a neutral face. But for the dictionary, the first frame is subtracted from the last τ_{trn} frames per video. The parameters are set as $\tau_{trn} = 8$, $\tau_{tst} = 8$, $\lambda_L = 10$ and $\lambda_G = 4.5$. 10 videos are randomly chosen for training and 5 for testing per class, due to only 17 contempt videos. Fig. 4.6 visualizes the recovery results after 600 iterations to convergences for sure. As matrix computations

[1]Contempt is discarded in [156, 157] but it is kept for the completeness of the experiment on CK+. See `https://github.com/eglxiang/icassp15_emotion` for face videos and programs.

Recognition Methods	Rates
DTGAN: [118]: joint mutually fine-tuning DTAN & DTGN	0.95
DTGN [118]: fully-connected DNN on key points	0.92
DTAN [118]: convolutional DNN (CNN) on raw images	0.91
3DCNN-DAP [162]: 3DCNN w/ deformable AU-like constraint	0.88
3DCNN [162]: 3DCNN (replicate model for action recog.)	0.78
H-CRF [160]: hidden CRF	0.88
ITBN [110]: spatio-temporal Bayes net	0.86
LD-CRF [161]: key points+latent dynamic (hidden) CRF	0.85
AAM+SVM [158]: key points (shape) & appearance + SVM	0.83
CLM+SVM [159] : key points + local model + SVM	0.74
RBF+SVM [160]: key points + multiclass SVM	0.72
DCS [156] on 6 classes: raw intensity + dual representation	0.81
SRC [157]: raw intensity + **neutral face provided**	0.67
SLR: raw intensity + joint-sparsity + low-rank interference	0.70
CHi-SLR: raw intensity + SLR + group sparsity	0.80

Table 4.2: Summary of total (average) recognition rates.

are mostly light, the iterative ADMM is parallelized using CUDA ($25 \sim 30$ fps).

For comparison, the image-based SRC [156, 157] model is replicated and assume the neutral face is assumed to be known. An expression component is represented by subtracting the neutral face (the first frame) from the last frame per video. 10 videos are also chosen for training and 5 for testing per class. Empirically, the optimal sparsity is 35% and it takes ≤ 100 iterations to converge.

Evaluation. Table 4.1 summarizes per-class performances. Top to down, the first three sub-tables present the confusion matrix for C-HiSLR, SLR and SRC, respec-

tively. Columns are predictions and rows are ground truths. The recognition rate is **0.80** with a std of 0.05 for C-HiSLR and **0.70** with a std of 0.14 for SLR. SRC [157] with neutral faces explicitly provided achieves an accuracy of **0.67** (std: 0.05). It is clear that the performance is boosted step by step among those three linear models of the same kind. Going bottom-up in Table 4.1, the fourth sub-table takes the diagonal from the confusion matrix to compare the true positive rate.

Along with the model name, the average recognition rates are given. They are also listed in Table 4.2. Most models' accuracy is in $(0.7, 0.9)$ except DTGAN [118], which mutually fine-tune an appearance-based CNN and a shape-based fully-connected network. While the intrinsic model is likely non-linear, expertise is needed in fine-tuning such a complex deep model. An adaptation of 3D-CNN to expression achieves just 0.78 [162]. Then, temporal modeling [110, 160, 161] surely helps but the gain over simply stacking frames is marginal (0.85 vs. 0.80).

Furthermore, the proposed C-HiSLR is more competitive than shape+SVM models such as CLM [159] (0.74), which makes sense because SRC works as a max-margin-like classifier. When the features are insufficient, adding non-linearity [160] does not help (RBF [160]: 0.72). Once combined with appearance, AAM+SVM [158] performs up to 0.83. C-HiSLR is comparable with the piecewise linear model DCS [156]) and is one of the simplest working models to approximate the problem's nature. An even simpler model SLR behaves unstably likely due to the inconsistent order of residuals. Note that the group sparsity may be violated due to viewpoints or poses (see Fig.

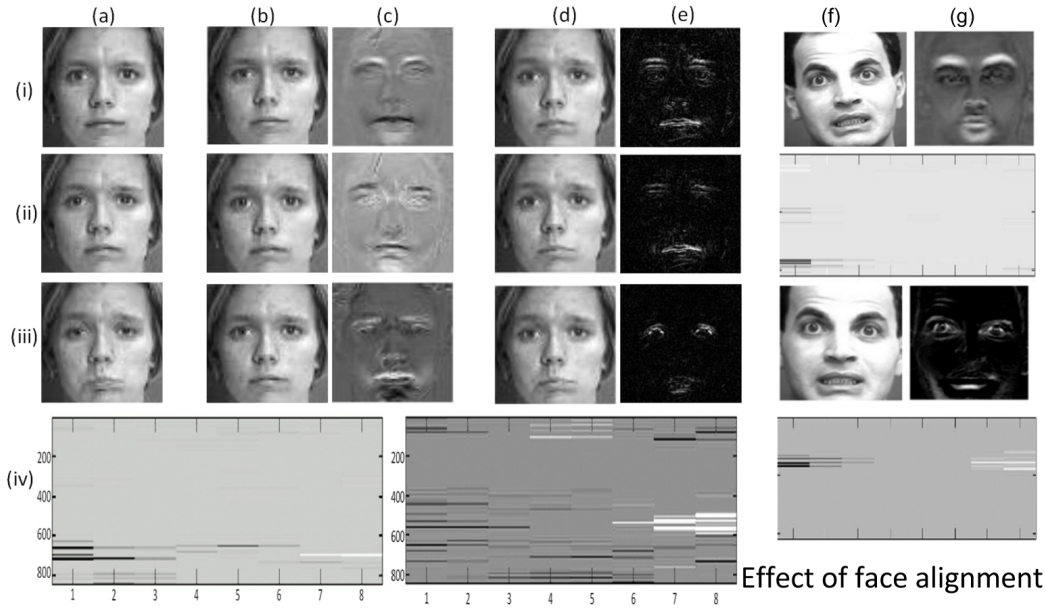


Figure 4.6: Effect of group sparsity. (a) is the test input $\mathbf{Y}$. (b)(c) are recovered $\mathbf{L}$ and $\mathbf{DX}$, given by **C-HiSLR** which correctly classifies (a) as *contempt*. (d)(e) are recovery results given by **SLR** which mis-classifies (a) as sadness. (i),(ii),(iii) denote results of frame #1, #4, #8 respectively, whereas (iv) displays the recovered $\mathbf{X}$ (left for C-HiSLR and right for SLR). In the right, (f) input face, (g) recovered $\mathbf{DX}$, (ii)(iV) recovered $\mathbf{X}$.

4.6) and alignment does not necessarily mean frontalization (see Fig. 4.2 and next section).

4.1.6.2 Facial action unit detection

To be pose-independent, the following experiments are performed on a profile view of MPI-VDB [2] containing 27 long video all with over 100 frames (1 video per class, see Fig. 4.2). From each video, the model samples 10 disjoint sub-videos each of which

[2]See `http://vdb.kyb.tuebingen.mpg.de` for raw data ad `https://github.com/eglxiang/FacialAU` for cropped face data.

SLR	0.87	0.78	**0.61**	0.92	**0.91**	**0.96**	**0.76**	0.03	**0.98**
CHi	**1.00**	**0.83**	**0.61**	**1.00**	0.70	0.85	**0.72**	**0.22**	**0.95**
SLR	0.51	0.76	0.85	**0.92**	0.93	**1.00**	**0.83**	**0.99**	0.73
CHi	**0.74**	**0.95**	**0.99**	**0.90**	**1.00**	**1.00**	**0.81**	**1.00**	**0.77**
SLR	0.57	**1.00**	**0.99**	**0.87**	0.27	**1.00**	**0.77**	**1.00**	0.62
CHi	**0.91**	**1.00**	0.89	0.62	**0.56**	**0.97**	**0.77**	**1.00**	**0.73**

Table 4.3: Accuracy on MPI-VDB. CHi means C-HiSLR.

contains 10 equally-spaced sampled frames. Different from Sec. 4.1.6.1, all frames are directly used without subtracting the first frame because the sub-videos do not start with neutral states. However, there underlie implicit neural states and presumably the proposed model is still valid. Then, 5 sub-videos are randomly sampled from the 10 for forming dictionary and the other 5 for testing (namely $\tau_{trn} = 5$ and $\tau_{tst} = 5$). In this way, the dataset is divided into a training set and a disjoint testing set both with 5 sub-videos per category.

Evaluation. When $\lambda_L = 15$, SLR's performance is shown in the left visulized confusion matrix in Fig. 4.7 with an average recognition rate of **0.80**. When $\lambda_G = 4.5$, C-HiSLR's performance is shown in Fig. 4.7-right with an average recognition rate of **0.84**. Both of their std are 0.04. As a result, the difference within 0.04 is treated as comparable in Table 4.3, which lists the per-class recognition rate for SLR and CHi-SLR, respectively. See Fig. 4.2 for the class index. For example, they both per-

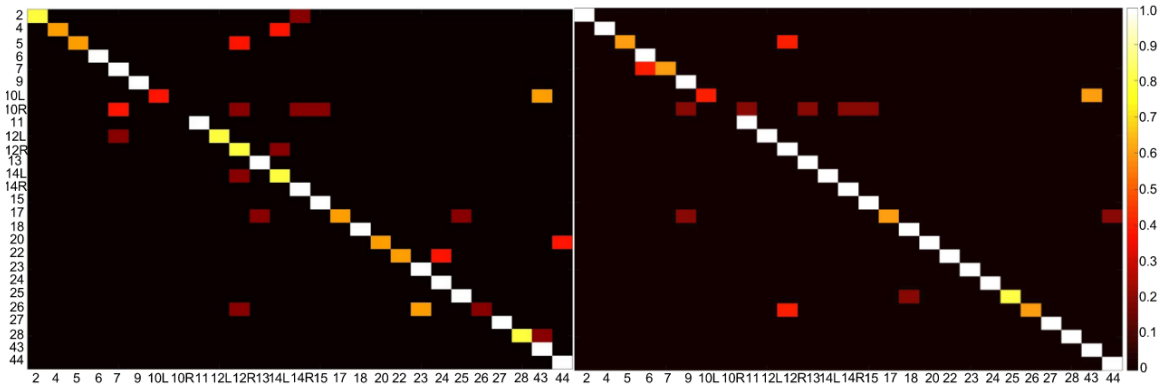


Figure 4.7: Confusion matrix for **SLR** (left) and **CHi-SLR** (right) on 27 AUs from MPI-VDB. SLR achieves a recognition rate of **0.80** while CHi-SLR achieves **0.84**. Best seen on the computer.

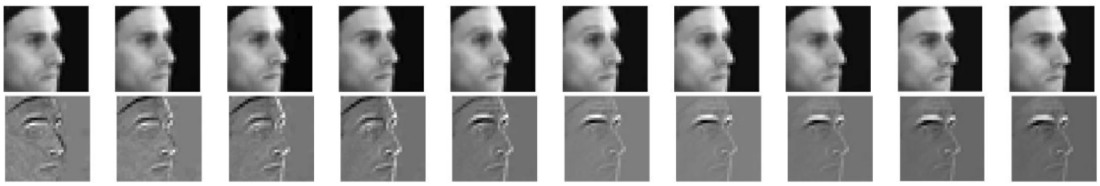


Figure 4.8: Example recovery (C-HiSLR). Top: **L**. Bottom: **AX**.

form poorly on the first row and the second last column (0.03 for SLR and 0.22 for C-HiSLR). Regarding the reason, it can be seen from Fig. 4.2 that the corresponding AU is 10R, which is the right upper lip raiser and confuses with right lip corner (12R). Similar for 13 (cheek puffer), 14R (right dimpler) and 15 (lip corner depressor) because they are all about lips. Note that related works discriminate over around 10 selected non-confusing AUs while in this experiment 27 AUs over the entire list of AUs are tested. Fig. 4.8 exemplifies the recovered neutral and AU components.

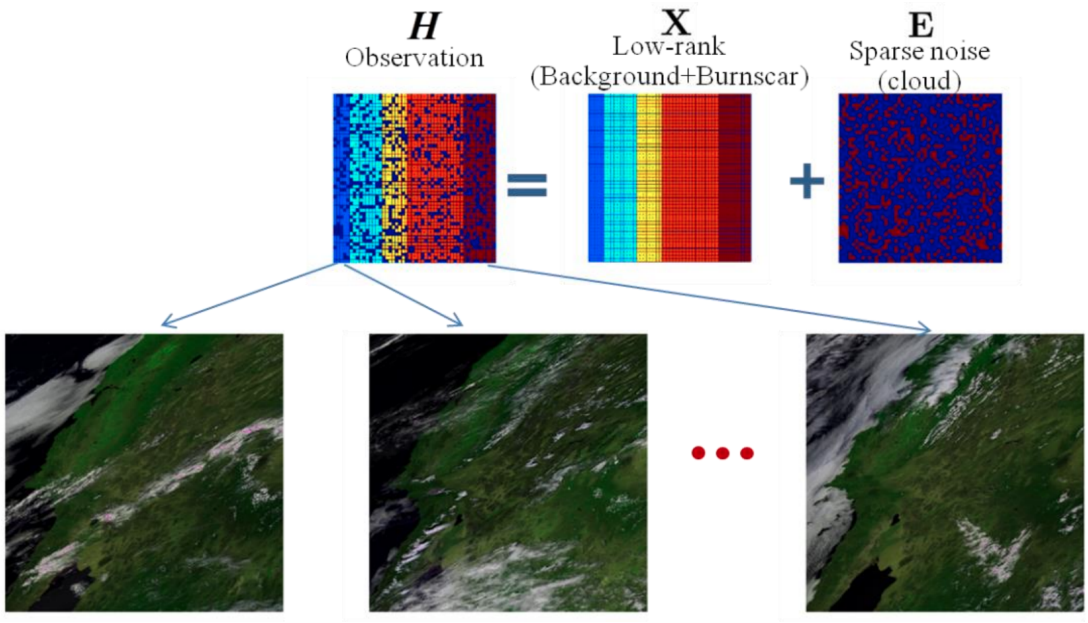


Figure 4.9: Cloud suppression via robust PCA.

4.1.6.3 Other examples: hyperspectral burnscar detection

In this section, the proposed C-HiSLR model is applied in hyperspectral imaging (HSI) data for the burnscar detection. RPCA has the capability of decomposing a matrix into an underlying low-rank structure and a sparse component. In the first step, this idea is applied in separating the cloud information from the processing HSI data. In the testing data set, each of the hyperspectral frames contains three main components of interest: the background content, the burning area and the cloud information. The background content of hyperspectral frames stays almost stationary while the burning area is also stable in consecutive days, hence the background-plus-burning content in each frame should be a low-rank component over time. Furthermore, the

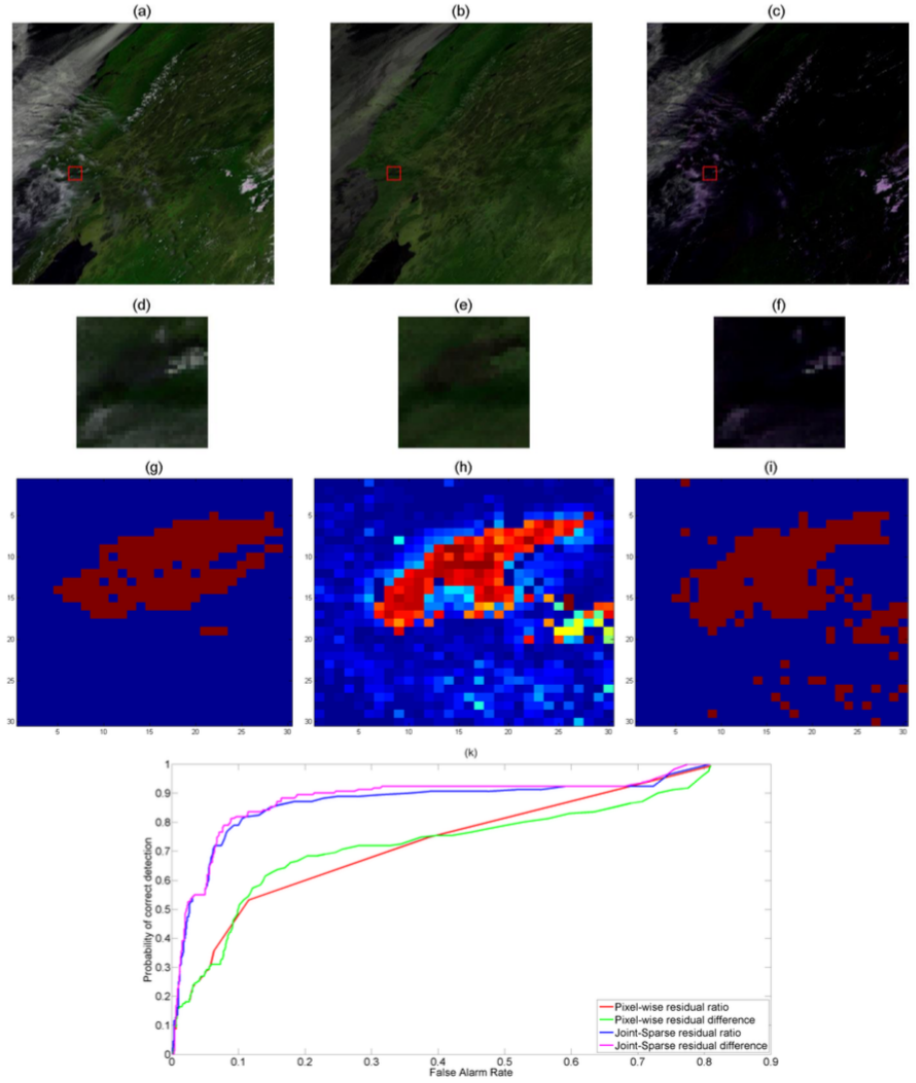


Figure 4.10: Burnscar detection result.

cloud information is changing very fast and may look completely different from any two consecutive days. Therefore, by vectorizing each spectral band of consecutive frames and concatenating into a big matrix $\mathbf{H}$ (*i.e.*, each seven columns of $\mathbf{H}$ are the seven bands of each hyperspectral frame in vector forms), $\mathbf{H}$ can be decomposed into a low-rank matrix $\mathbf{X}$ and a sparse component $\mathbf{E}$ using the RPCA algorithm where $\mathbf{X}$

contains the background-plus-burning information while **E** captures all cloud content (Fig. 7.1). Matrix **X** is then used as the input of the detection step to determine the burning area while **E** can be utilized to decide the area that cloud occurs as well as the cloud level presenting in every pixel (which may be used to robustify the detection step).

Fig. 4.10 shows the detection results. (a) shows the original HSI image in RGB mode. (b) shows the output low-rank component of RPCA. (c) shows the output sparse component of RPCA (cloud). (d), (e) and (f) show the zoom-in patches of HSI images within the burnscar testing region (red rectangle region) shown in (a), (b) and (c), respectively. (g) shows the ground truth burn map within the testing patch. (h) shows the output residual difference of joint-sparsity detection. (i) shows the output burnscar detection obtained by thresholding. (h) shows the receiver operating characteristic (ROC) curves of pixel-wise and joint-sparsity target detection.

After suppressing the cloud information in every hyperspectral frame, the next step is to automatically classify every pixel if it belongs to a burnscar or non-burnscar area using a sparsity-based representation model. Furthermore, the target detection performance is robustified via a spatial-temporal joint sparsity model which derives from the observation that neighborhood pixels from the same hyperspectral image as well as previous and next hyperspectral frames usually have similar spectral characteristics. Thus, by representing nearby pixels simultaneously in a batch processing, another level of robustness can be brought into the burnscar detection result. The

HSI sparsity model is based on the assumption that the spectral characteristic of each pixel approximately lies in the low-dimensional subspace that is spanned by the training samples of the same class presenting in that pixel.

With the observation that small neighborhood HSI pixels usually belong to the same materials, a joint sparsity model is further proposed by enforcing them to have the same sparsity support of the training samples. If an observed pixel is defined as background, it is very likely that the nearby pixels in the same frame are also defined as background and should belong to the same class of materials as the testing observation. Not only that, the pixels at the same locations in the previous and next frames should also share the same materials. Moreover, the area that a burn appears should also smooth. Consequently, if an observed pixel is supposed to be in a burning area, the pixels in its small neighborhood in both spatial and time domain should also be classified as being burned. Therefore, they can be approximated by a linear sparse combination from the same few samples in the target-background dictionary.

4.1.7 Summary

In this section, an expression-identity decoupled linear model is proposed to learn a representation of expression, unlike [157] requiring neutral faces as inputs and [156] generating labels of the identity and expression as mutual by-products yet with extra efforts. Our contribution are two-fold. First, the proposed model retains the separation of the neutral face and the sparse representation of the expression component

jointly. Second, both the atom-wise and group-wise joint sparsity over channels are enforced to induce consistent classification over frames. For the CK+ dataset, C-HiSLR's performance on raw faces is way more competitive than SRC given neutral faces and shape+SVM methods. The model is applied in recognizing actions units with limited training data achieving decent performances, which should be generalizable to human action dataset such as NTU RGB+D with 3D skeletons. While it needs work to ensure such properties in the feature space, there exist temporal models correlating CNN features over time.

4.2 Temporal recursive matching pursuit

In this section, the facial expression recognition problem is formulated as a video Sparse Representation based Classification (SRC) with Long Short-Term Memory (LSTM) mechanism. The proposed sparse coding with temporal modeling using LSTM is designed to detect salient action units (AUs) over the time of repeated expressions, yet with a much low computation cost credited to the selective AUs.

4.2.1 Coding pose-specific expression with LSTM

This section models an implicit latent representation $\mathbf{X} \in \mathbb{R}^{n \times \tau}$ of an input test face $\mathbf{Y} \in \mathbb{R}^{d \times \tau}$ as a sparse linear combination of prepared fixed training emotions $\mathbf{D} \in \mathbb{R}^{d \times n}$: $\mathbf{Y} = \mathbf{DX}$, where the dictionary matrix $\mathbf{D}$ is an arrangement of all sub-matrices $\mathbf{D}_{[j]}$, $j = 1, ..., \lfloor \frac{n}{\tau} \rfloor$. Notably, n is assumed to be much larger than d and rank($\mathbf{D}$)= d. Namely, our task is to sequentially find a small subset (*i.e.*, basis set) of columns from $\mathbf{D}$ for $\mathbf{X}$, the Multiple Measurement Vectors (MMV).

4.2.1.1 Recursive Matching Pursuit Using RNN

As the basis set is built up by adding a single column vector at a time, this formulation denotes the residual matrix after the p-th iteration by $\mathbf{R}^{(p)} \in \mathbb{R}^{n \times \tau}$ with $\mathbf{R}^{(0)} = \mathbf{Y}$. The i-th column of $\mathbf{Y}^{(p)}$ is denoted by $\mathbf{y}_i^{(p)}$. The indices of the p vectors selected are stored in the index set denoted by $\mathbb{I}^{(p)}$; where $\mathbb{I}^{(p)} = \{k_1, k_2, ..., k_p\}$ and $\mathbb{I}^{(0)} = \emptyset$. The selected columns vectors are stored in a matrix $\mathbf{S}^{(p)} = [\mathbf{d}_{k_1}, \mathbf{d}_{k_2}, ..., \mathbf{d}_{k_p}]$

and $\mathbf{S}^{(0)} = \emptyset$. The orthogonal projection matrix onto the column space of $\mathbf{S}^{(p)}$ is denoted by $\mathbf{P}_{\mathbf{S}^{(p)}}$ and its orthogonal complement $\mathbf{P}^{\perp}_{\mathbf{S}^{(p)}} = (\mathbf{I} - \mathbf{P}_{\mathbf{S}^{(p)}})$ and $\mathbf{P}_{\mathbf{S}^{(0)}} = \mathbf{0}$, $\mathbf{P}^{\perp}_{\mathbf{S}^{(0)}} = \mathbf{I}$ where $\mathbf{I}$ is an identity matrix and $\mathbf{0}$ is a zero matrix.

Algorithm 3: RMP using Recurrent Neural Network.

function $\mathbf{X}$ = RMP-RNN($\mathbf{Y}, \mathbf{D}, resMin$);

Input : measurement matrix $\mathbf{Y} \in \mathbb{R}^{d \times \tau}$, dictionary matrix $\mathbf{D} \in \mathbb{R}^{d \times n}$, minimum Frobenius norm $resMin$, trained RNN model.

Output: sparse-codes matrix $\mathbf{X} \in \mathbb{R}^{n \times \tau}$

Initialization $\mathbf{D}^{(0)} = \mathbf{D}$, $\mathbf{X}^{(0)} = \mathbf{0}$, $\mathbf{R}^{(0)} = \mathbf{Y}$;

while $i \leq \tau$ *and* $\|R\|_F \leq resMin$ **do**

- $p \leftarrow p + 1$
- $\mathbf{r}_i^{(p)} \leftarrow \frac{\mathbf{r}_{i-1}^{(p)}}{\max\left(|\mathbf{r}_{i-1}^{(p)}|\right)}$
- $\mathbf{h}_i \leftarrow \text{RNN}(\mathbf{r}_i^{(p)}, \mathbf{h}_{i-1}, \mathbf{c}_{i-1})$
- $\mathbf{c}_i \leftarrow \text{softmax}(\mathbf{U}\mathbf{h}_j)$
- $k_p \leftarrow \text{Support}(\max(\mathbf{c}))$
- $\mathbb{I}^{(p)} \leftarrow \mathbb{I}^{(p-1)} \cup k_p$
- $\mathbf{S}^{(p)} \leftarrow [\mathbf{S}^{(p-1)}, \mathbf{d}_{k_p}]$
- $\mathbf{P}_{\mathbf{S}^{(p)}} \leftarrow \mathbf{P}_{\mathbf{S}^{(p)}, k_p}$ by Eqn. (4.13)
- $\mathbf{d}_i^{(p)} \leftarrow$ Eqn. (4.15)
- $\mathbf{x}_i^{(p)\mathbb{I}} \leftarrow \left(\mathbf{S}^{(p)}\right)^{\dagger}\mathbf{y}_i$
- $\mathbf{r}_i^{(p)} \leftarrow$ Eqn. (4.16)

The basic idea of the Recursive Matching Pursuit (RMP) algorithm is the pursuit of the matching p-th basis vector conceptually involves solving $(n-p+1)$ order recursive least squares problems and selecting the vector that reduces the residual the most.

Then, $\mathbf{d}_k^{(0)} = \mathbf{d}_k (\forall k = 1, ..., n)$ is initialized and a column of $\mathbf{D}$ indexed by

$$k_p =_k \Big(\sum_{i=1}^{\tau} \Big| \big(\mathbf{d}_k^{(p-1)}\big)^T \mathbf{r}_i^{(p)} \Big|^2 \Big/ \|\mathbf{d}_k^{(p-1)}\|^2 \Big). \tag{4.12}$$

are chosen. As $\mathbf{S}^{(p)}$ is augmented, the next step is updating $\mathbf{P}_{\mathbf{S}^{(p)}}$ by

$$\mathbf{P}_{\mathbf{S}^{(p)},k_p} = \mathbf{P}_{\mathbf{S}^{(p)}} + \mathbf{q}^{(p)} \big(\mathbf{q}^{(p)}\big)^T \tag{4.13}$$

where

$$\mathbf{q}^{(p)} = \frac{\mathbf{d}_{k_p}^{(p-1)}}{\|\mathbf{d}_{k_p}^{(p-1)}\|}. \tag{4.14}$$

Now the columns of $\mathbf{D}$ and $\mathbf{R}$ are projected to the column space of $\mathbf{S}^{(p)}$ and there is

$$\mathbf{d}_i^{(p)} = \mathbf{P}_{\mathbf{S}^{(p)}}^{\perp} \mathbf{d}_i^{(p-1)} = \mathbf{d}_i^{(p-1)} - \Big(\big(\mathbf{q}^{(p)}\big)^T \mathbf{d}_i^{(p-1)} \Big) \mathbf{q}^{(p)}, \tag{4.15}$$

$$\mathbf{r}_i^{(p)} = \mathbf{P}_{\mathbf{S}^{(p)}}^{\perp} \mathbf{r}_i^{(p-1)} = \mathbf{r}_i^{(p-1)} - \Big(\big(\mathbf{q}^{(p)}\big)^T \mathbf{r}_i^{(p-1)} \Big) \mathbf{q}^{(p)}, \tag{4.16}$$

respectively, $\forall i = 1, 2, ..., \tau$. Note that no orthogonal projection is employed in the updates. Selection of a column of $\mathbf{D}$ corresponds to selecting a nonzero row of $\mathbf{X}$. The nonzero rows of $\mathbf{X}$ form $\mathbf{X}^{\mathbb{I}}$ which is given by $(\mathbf{S}^{(p)})^{\dagger}\mathbf{Y}$ where $\dagger$ denotes pseudo-inverse.

4.2.1.2 RNN Using Long Short-Term Memory (LSTM)

$$\mathbf{h}_i = \mathbf{o}_i * \tanh(\mathbf{f}_i * \mathbf{c}_{i-1} + \mathbf{e}_i * \mathbf{c}_i) \tag{4.17}$$

where the information memorizing cell is given by

$$\mathbf{c}_i = \tanh(\mathbf{W}_c[\mathbf{h}_{i-1}, \mathbf{r}_i] + \mathbf{b}_c), \tag{4.18}$$

the vector of information forgetting gates is given by

$$\mathbf{f}_i = \sigma(\mathbf{W}_f[\mathbf{h}_{i-1}, \mathbf{r}_i] + \mathbf{b}_f), \tag{4.19}$$

the vector of information entering gates is given by

$$\mathbf{f}_e = \sigma(\mathbf{W}_e[\mathbf{h}_{i-1}, \mathbf{r}_i] + \mathbf{b}_e), \tag{4.20}$$

and the vector of information outputing gates is given by

$$\mathbf{f}_o = \sigma(\mathbf{W}_o[\mathbf{h}_{i-1}, \mathbf{r}_i] + \mathbf{b}_o). \tag{4.21}$$

4.2.2 Baseline: Simultaneous Recursive Matching Pursuit

Sparse coding has its root in neuroscience and has been well exploited in harmonic analysis, signal processing and compressive sensing. Matching Pursuit dates back to 1993 in [89] and Recursive Matching Pursuit for sparsely coding Multi-Measurement Vectors can be traced back to 1998 in [90].

Algorithm 4: Baseline: simultaneous recursive matching pursuit (SRMP).

function $\mathbf{X} = \text{RMP}\ (\mathbf{Y}, \mathbf{D}, resMin)$;

Input : measurement matrix $\mathbf{Y} \in \mathbb{R}^{d \times \tau}$ and minimum Frobenius norm $resMin$ as stopping criterion.

Output: Approximation matrix $\mathbf{A} \in \mathbb{R}^{d \times \tau}$ and a set Λ_p containing p indices where p is the number of iterations.

Initialization $\mathbf{D}^{(0)} = \mathbf{D}$, $\mathbf{X}^{(0)} = \mathbf{0}$, $\mathbf{R}^{(0)} = \mathbf{Y}$;

while $i \leq \tau$ *and* $\|R\|_F \leq resMin$ **do**

- $p \leftarrow p + 1$
- $k_p \leftarrow$ Eqn. (4.12)
- $\mathbb{I}^{(p)} \leftarrow \mathbb{I}^{(p-1)} \cup k_p$
- $\mathbf{S}^{(p)} \leftarrow [\mathbf{S}^{(p-1)}, \mathbf{d}_{k_p}]$
- $\mathbf{x}_i^{(p)\mathbb{I}} \leftarrow \left(\mathbf{S}^{(p)}\right)^{\dagger} \mathbf{y}_i$
- $\mathbf{d}_i^{(p)} \leftarrow$ Eqn. (4.15)
- $\mathbf{r}_i^{(p)} \leftarrow$ Eqn. (4.16)

Recurrent Neural Network (RNN) is deep in time and thus can be treated as a deep neural network with the issue of vanishing gradients. Long Short-Term Memory network [92] dating back to 1997 is a type of RNN that gets around of vanishing gradient problem. Note that there also exists a type of network called Recursive Neural Network [93] which is generalized RNN with a deep structure of a skewed tree. There are connections among all those models and hidden Markov models.

Both a feed-forward multilayer neural network and a hidden Markov model can be seen as a directed acyclic graph with hidden nodes. Both a recurrent neural network and a hidden Markov model map a sequence of inputs to a sequence of outputs via a sequence of hidden states. Long Short-Term Memory learns a function of inputs and hidden states using the perceptron-like network with gating weights further learned using perceptrons.

4.3 Conclusion

In this chapter, the C-HiSLR representation model is designed for emotion recognition, unlike [157] requiring neutral faces as inputs and [156] generating labels of identity and emotion as mutual byproducts with extra efforts. This work mainly comes from a series of questions. First, how to get rid of the neutral face? Second, how to recover the low-rank and sparse components? Third, can such an approximate explicit Principal Component Pursuit step be avoided? Finally, should video frames be treated separately or simultaneously? And should a class-wise sparsity separately or enforce a group sparsity collaboratively be enforced?

Our contribution is two-fold. First, emotions are not recovered explicitly. Instead, frames are treated simultaneously and the low-rank neutral face are subtracted implicitly. Second, the label consistency is preserved by enforcing atom-wise as well as group sparsity. For the CK+ dataset, C-HiSLRs performance on raw data is com-

parable with SRC given neutral faces, which verifies that emotion is automatically separable from expressive faces as well as sparsely representable. Future works include handling misalignment [166] and incorporating dictionary learning [82]. This work has been published in our conference paper [100] and journal paper [167].

In addition, this chapter also shows how to extend the across-frame model into a temporal model using the LSTM type of algorithm for recursive matching pursuit during the optimization. The problem of facial expression recognition is formulated as a video Sparse Representation based Classification (SRC) with LSTM mechanism, which is applicable for human actions yet requiring a careful design of sparse representation due to possible changing scenes. This extended algorithmic work has been published in [168].

Chapter 5

Temporal Representation of Egocentric Videos via Motion Models

Egocentric videos are also called first-person videos, normally captured by a moving camera (say, phone cameras) or a body-mounted camera (say, ©GoPro cameras, ©Google glasses, ©Snap spectacles, Augmented Reality headset cameras). For example, Fig. 8.2 shows the keyframes of an egocentric video captured using a head-mounted GoPro camera. In other cases, as shown in Fig. 5.2, a moving device instead of a person becomes the camera platform (say, dash cameras, self-driving cars' cameras, flying drones' cameras, servant robot's cameras, endoscopic cameras). For example, Fig. 5.3 shows a screenshot of a video captured by a dashcam and Fig.

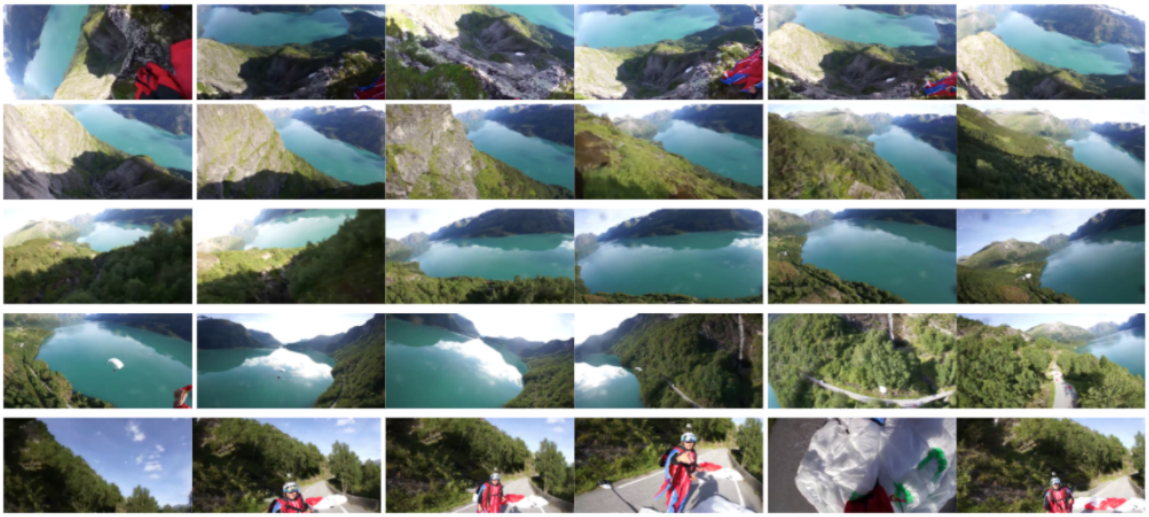


Figure 5.1: Keyframes extracted from a video collected using a head-mounted GoPro camera in the SumMe dataset.

5.4 shows screenshots of a video captured by a servant robot camera.

One of key aspects of temporal dynamics is the motion and object movement. For egocentric videos, the movements come from both the camera movement and the object movement - the so-called dynamic scenes as shown in Fig. 5.5.

For example, the camera motion estimation is a standard yet critical step to endoscopic visualization. It is affected by the variation of locations and correspondences of features detected in 2D images. Feature detectors and descriptors vary, though one of the most widely used remains SIFT. Practitioners usually also adopt its feature matching strategy, which defines inliers as the feature pairs subjecting to a global affine transformation. However, surfaces are non-planar in endoscopic videos. The first question to be answered is if it is more suitable to cluster features into multiple groups. Then within each group, the same transformation can still be enforced as

Figure 5.2: Various platforms that host mobile cameras.

Figure 5.3: A screenshot of a video captured by a dashcam.

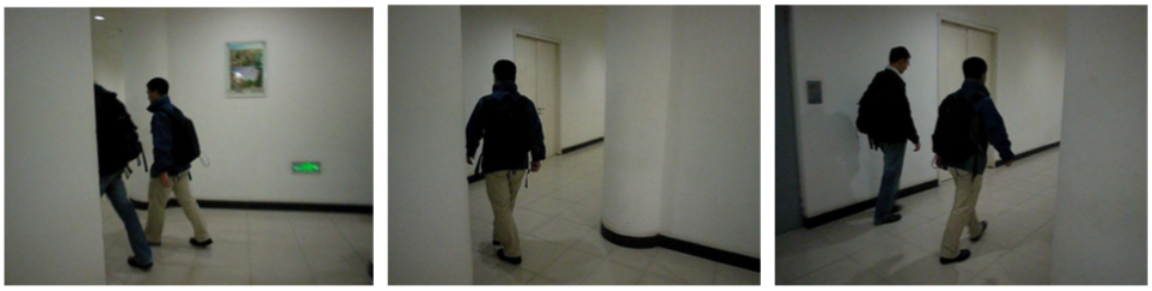


Figure 5.4: Screenshots of a video captured by a camera on a servant robot that follows the target person.

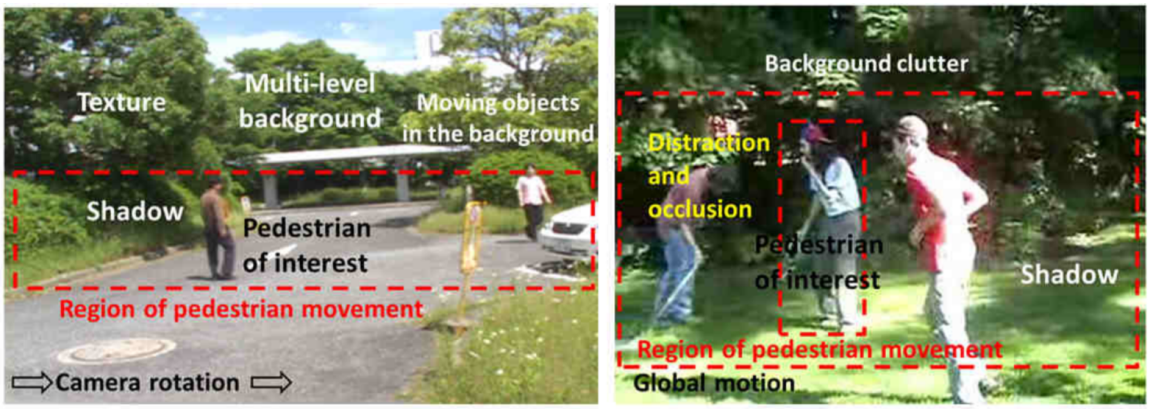


Figure 5.5: Characteristics of humans in dynamic scenes.

done in SIFT. Such a multi-model idea has been recently examined in the Multi-Affine work, which outperforms Lowe's SIFT in terms of re-projection error on minimally invasive endoscopic images with manually labeled ground-truth matches of SIFT features. Since their difference lies in matching, the accuracy gain of estimated motion is attributed to the holistic Multi-Affine feature matching algorithm. But, more concretely, the matching criterion and point searching can be the same as those built in SIFT. It is arguable that the real variation is only the motion model verification. Either a single global motion model or a group of multiple local ones is enforced. This section investigates how sensitive the estimated motion is affected by the number of motion models assumed in feature matching. While the sensitivity can be analytically evaluated, an empirical analysis is presented in a leaving-one-out cross validation (LOOCV) setting without requiring labels of ground-truth matches. Then, the sensitivity is characterized by the variance of a sequence of motion estimates. A series of quantitative comparison is presented such as accuracy and variance between the Multi-Affine motion models and a global affine model used in SIFT.

In the second example work, a robust tracker is designed and named MIL-PF. It is because that it combines the merits of the Multiple Instance Learning tracker and and the Particle Filter tracker. Online updated boosting is not robust but enable recovering from the error. Bayesian filter tracker cannot track the target accurately but does not drift far away. It is sensible to combine an online updated appearance-based boosting tracker with the motion model in the particle filter.

5.1 Global motion estimation: robust feature matching and its sensitivity

This section focuses on the **motion estimation** which interacts with **feature matching**, particularly in the scenario of endoscopic videos, as shown in Fig. 5.6. While the global camera motion is estimated from matched features, first a preliminary motion model is needed to verify the feature matches. Subsequently, [169] shows the limitations of image-based tracking alone can be overcome by employing an Electro-Magnetic (EM) tracker, which provides a rough location. EM tracking can correct drifting, while frame-by-frame tracking-by-matching gives a refined location. Lastly, reconstruction can follow a point cloud generation by either simple triangulation [170, 171] or bundle adjustment [172, 173], or surface rendering methods using shading [174] and specularities [175].

In such a pipeline of 3-D visualization, the camera motion estimation is critical to the final accuracy. It is standard to estimate the global motion once feature matches are available. Eight-point algorithm [171] or the relaxed five-point algorithm [176] give quite similar estimates. [177] can even handle the wide-baseline problem. Then, the variation comes from previous steps such as feature detection, description and matching. The Scale Invariant Feature Transform (SIFT) [178] is invariant to image scaling and rotation, and partially invariant to changes in illumination and 3D camera viewpoint. It is a good idea to fix the detector and descriptor to be SIFT and then

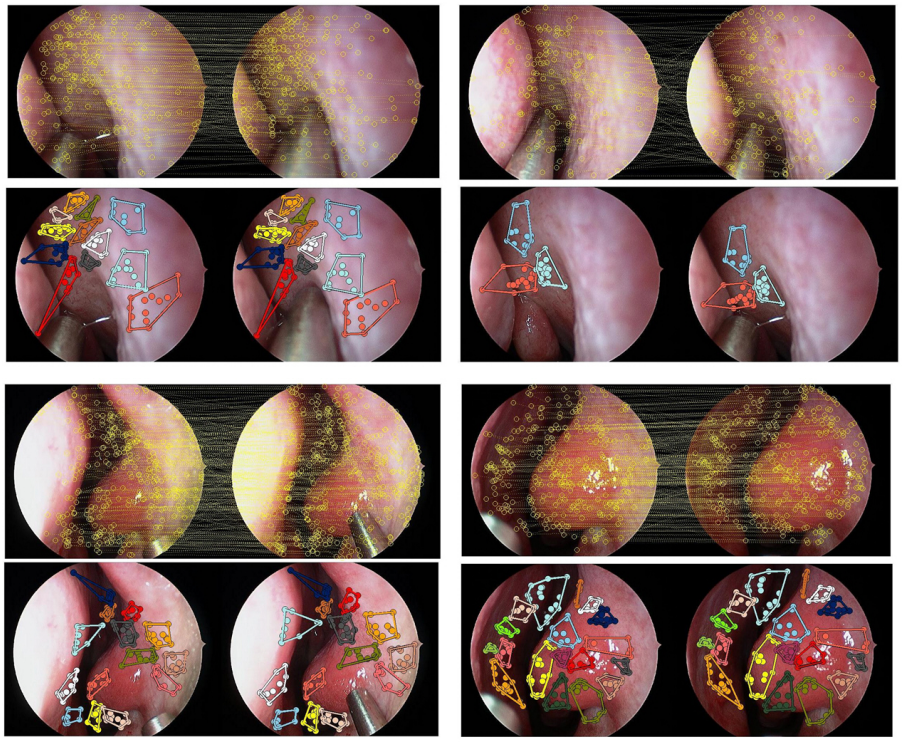


Figure 5.6: Examples of SIFT's global-affine vs. HMA's multi-affine model. In each pair, the top row shows SIFT's result, in which line crossings imply mismatches. The bottom row shows HMA's result, in which different components are displayed in different color.

concentrate on examining how sensitive the estimated motion is affected by the feature matching.

The matching algorithm built in the original SIFT [178] is compared with the state-of-the-art Multi-Affine matching [179–181] (typically the Hierachical Multi-Affine (HMA) [180]). In detail, feature matching consists of deciding the matching criterion, searching a similar feature and verifying if the matches agree with the motion model [182]. Firstly, SIFT's matching strategy is thresholding the ratio of nearest/2nd-nearest Euclidean distance in the feature space. HMA follows that as

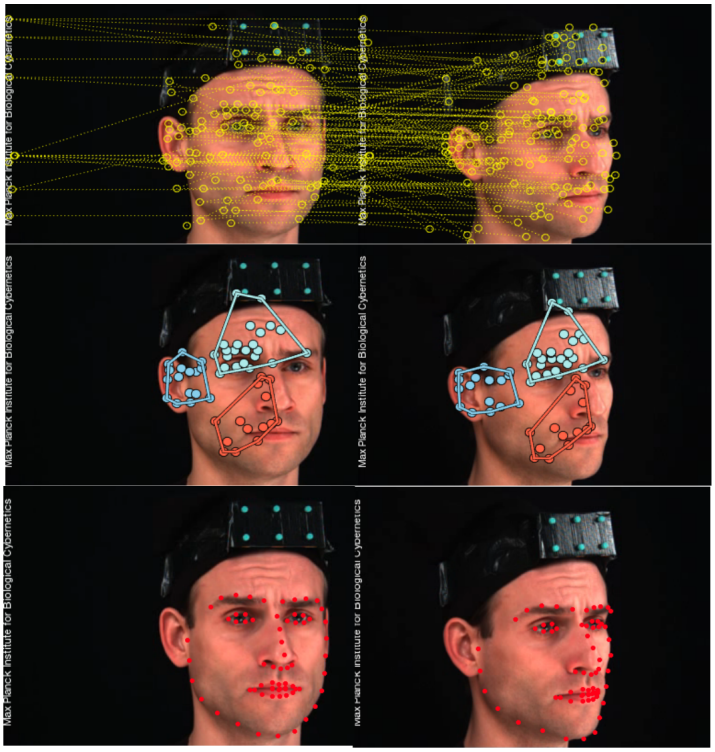


Figure 5.7: Feature matching. Top: single-model RANSAC. Middle: multi-model RANSAC. Third: prior landmark model, used as ground truths to evaluate the matching quality.

well. Secondly, the variation of searching algorithms highlighted in [180] is about efficiency. Thirdly, matching is normally expected to be robust to outliers and deformation. Therefore, a model verification step aims to check the agreement between each feature and the motion model. However, HMA and the SIFT differ in the number of motion models. It is entirely possible to replace the affine model built in SIFT with any linear or non-linear model. However, it is reasonable to fix it to be affine and then investigate if multi-model is really superior over single-model for feature matching.

As a result, this section attempts to illustrate how sensitive the estimated motion

is affected by the number of motion models used in feature matching. Surely the sensitivity of estimated motion $[\mathbf{R}, \mathbf{t}]$ can be approximately analyzed by applying a direct differentiation of the epipolar constraint [183] and linearizing the motion displacement $[\Delta\mathbf{R}, \Delta\mathbf{t}]$. Now it is necessary to explicitly write down the correlation between the covariance of $[\mathbf{R}, \mathbf{t}]$ and the variance of the matched feature pairs [184]. However, it is still significant to verify such an approximation with the empirical sensitivity by estimating the motion a number of times. In this way, the accuracy of feature matching can also be computed in terms of inlier ratio and the accuracy of estimated motion in terms of re-projection error.

Notably, while our analysis is presented for endoscopic images, there is no restriction on the content of images that multi-modal feature matching can be applied and analyzed. The fact is that it works way better than single-modal feature matching for textureless non-planar surfaces, which are widely existed among deformable objects such as organs in endoscopic images and faces in natural images (see Fig. 5.7).

5.1.1 Theoretical property of multi-model feature matching

This section investigates how the number of motion models affects feature matching, as all other algorithmic steps are fixed. The input to a feature matching algorithm are features and a parameterized motion model, while the outputs are feature pairs

and motion model parameters. Let the features be fixed as SIFT features and the motion model simply be an affine transformation based model, which can be either a single affine model or a mixture of multiple affine models. Once the motion model parameters are estimated, a feature matching algorithm consists of searching a similar feature according to a certain matching criterion and then verifying if the matches agree with the motion model. the matching criterion can be fixed as thresholding the ratio between the nearest and the second nearest Euclidean distance in the feature space. As a convention, the k-d tree searching algorithm is used. The motion model verification uses the estimated motion model to validate the consistency of feature pairs in terms of the re-projection error. Those matches that exhibit a re-projection error larger than a threshold are considered outliers (*i.e.*, mismatches). On the other hand, inliers are defined as the pairs subject to the refined motion model.

Note that a single affine transformation can be characterized using 6 parameters: 4 for the linear transformation and 2 for the translation. As a result, the linear system is solvable by Least Squares as long as there are over 3 feature pairs *i.e.*, 6 equations considering 2-D coordinates). However, the feature pairs may be contaminated by mismatches. Now each feature pair is treated as a data sample, then a mismatch is an outlier. A typical robust parameter estimation algorithm is Random Sample Consensus (RANSAC), which iteratively use a minimal number of pairs needed to re-estimate the model in terms of feature matching. However, RANSAC is widely known as a single-model approach. When the data are drawn from multiple models,

RANSAC may fail to find any one of the models. Now, given n data points drawn from m models, a straightforward algorithm is to run RANSAC sequentially. Each iteration of these methods selects one randomly sampled model maximizing either the number of inliers or some similar threshold-based measure. Here the minimum number of samples are presented. These samples are needed by the multi-modal RANSAC.

Proposition 3. RANSAC needs over $\log(m)(\frac{m}{w})^n \log(\frac{1}{1-p})$ samples to be guaranteed to fit n data points to the m models with an inlier ratio w and a probability p.

Proof. It is assumed that the percentage of inliers to one model is $\frac{w}{m}$. One mode is fitted at a time and its fitted points are removed from the total point set. In each sampling step, n points are taken and checked if they belong to a model. Namely, if the n points in the sample belong to a model, then the points associated to this model are removed. And RANSAC now looks for other models. If the n points in the sample do not belong to a model, Sampling is kept going until fitting a model. Then the model is fitted in an iterative manner and reduce the total point set until generating the m models.

Suppose, $m-l$ models have already been defined, and let w_l be the proportion of inliers when there are still l models to fit (therefore, $w_m = w$). The probability that all chosen n points belong to one of the remaining l models is $w_l(\frac{w_l}{l})^n$. Therefore, to have at least one valid model with probability p after k_l sampling, the following will

be required,

$$1 - (1 - l(\frac{w_l}{l})^n)^{k_l} \geq p \tag{5.1}$$

As a result,

$$k_l \geq \frac{\log(1-p)}{\log(1 - l(\frac{w_l}{l})^n)} \tag{5.2}$$

Using the approximation $\log(1 - l(\frac{w_l}{l})^n) \approx -l(\frac{w_l}{l})^n$, there is

$$k_l \geq \frac{\log(1-p)}{-l(\frac{w_l}{l})^n} = \frac{1}{l}(\frac{l}{w_l})^n \log(\frac{1}{1-p}) \tag{5.3}$$

Each time a model is defined and removed from the total point set, some inliers are removed, so the proportion between inliers to outliers decrease. In fact, it is easy to check that the proportion of inliers w_l satisfies $w_l \leq \frac{l}{m}w$. Using this inequality, there exists

$$k_l \geq \frac{1}{l}(\frac{m}{w})^n \log(\frac{1}{1-p}) \tag{5.4}$$

The total number of samples in the construction of all the m models is given by $k = k_m + k_{m-1} + ... + k_1$. Assuming that the l-th model was indeed found in at most $\frac{1}{l}(\frac{m}{w})^n \log(\frac{1}{1-p})$, then there is

$$k \geq \sum_{l=1}^{m} k_l \geq \sum_{l=1}^{m} \frac{1}{l}(\frac{m}{w})^n \log(\frac{1}{1-p}) \approx \log(m)(\frac{m}{w})^n \log(\frac{1}{1-p}) \tag{5.5}$$

The number of samples required is exponential in n and just polynomial in m.

In this section, the motion estimation's sensitivity, which is captured in the variance of the estimated rotation [173], is quantitatively analyzed. The basic idea is to

generate a sequence of estimates by Leaving-One-Out Cross Validation (LOOCV). In each trial, one feature pair is used for querying and the remaining for estimating the motion. In this way, it can be validated regarding how the holistic motion estimation pipeline generalizes such as robustness without requiring many batches of data. The less sensitively a model behaves on real data, the more robust it is.

5.1.2 Empirical sensitivity analysis: cross-validating estimated motion

For feature matching, the inputs are features and a motion model, while the output are the feature pairs and motion model parameters. For motion estimation, the inputs are feature pairs and the output are motion model parameters. Note that the two motion models can be different. The former is a preliminary model assumed to characterize a reasonable transformation between two images, which are not arbitrary two images though. They characterize adjacent views which approximately capture the same scene from a monocular camera. Thus, the latter model subjects to a more strict geometric constraint called the epipolar constraint [171]. It is normally affine as well, with a rotation and a translation. And its estimation method is quite standard such as the five-point algorithm [176]. The only variation is the design of the preliminary motion model. SIFT uses a global affine model while HMA uses a local affine model for each component, which is obtained through a hierarchical K-means

clustering and expected to represent a plane [181]. The way to estimate the model parameters is once again rather standard. In both cases, the model parameters are computed by solving a linear system from feature pairs for the model parameters. Then, they are refined by verifying the agreement between the raw feature pairs and the parameterized model based on some robust fitting methods, such as the Hough transform that SIFT uses and the RANdom SAmple Consensus (RANSAC). RANSAC is used for both HMA and SIFT. Its basic idea is to iteratively use a minimal number of pairs needed to re-estimate the model [171]. **Inliers** are defined as the pairs subjecting to the refined motion model. While inliers are detected separately for each group in HMA, its holistic inlier ratio with SIFT will be compared. Also note that one variation in our sensitivity analysis is the representation of a rotation. Instead of quaternions [185], another way is to use Euler angles [186]. When rotation angles are small, both ways can be an alternative to the rotation matrix [187]. Finally, the exact algorithm is elaborated in Algorithm 5.

Algorithm 5. Sensitivity analysis of motion estimation by LOOCV.

for $k = 1...FrmNum$

Form a candidate pool: detect SIFT key points and compute SIFT features.

Match features: fit feature pairs to a single-affine **or** multi-affine model.

Generate an inlier set: perform RANSAC on the matched key point pairs.

Rectify images considering radial distortion.

Convert image coordinates to World's coordinates using intrinsic parameters.

if $MatchedInlierNum > 4$

for $trial = 1...MatchedInlierNum$

Leave the $trial$-th key point pair $(\mathbf{p}_{left}^{query}, \mathbf{p}_{right}^{query})$ out as a query.

Estimate the essential matrix $\mathbf{E}$ using the remaining pairs.

Factorize $\mathbf{E}$ into a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$.

Convert $\mathbf{R}$ to a quaternion **or** yaw, pitch and roll angles rx, ry, rz.

Compute square of re-projection error for the held-out query point:

$$sqErr = \|(\mathbf{R} * \mathbf{p}_{left}^{query} + \mathbf{t}) - \mathbf{p}_{right}^{query}\|_2^2$$

end for

Compute the mean and standard deviation of a sequence of $sqErr$.

Compose a $\mathbf{R}_{mean}$ from $mean(quaternion)$ **or** $mean(rx), mean(ry), mean(rz)$.

for $trial = 1...MatchedInlierNum$

$\mathbf{R}_{mean} * \mathbf{R}^{-1}$ is approximately a skew-symmetric matrix $skew(\alpha, \beta, \gamma)$

end for

Compute the standard deviations of rotation angles α, β, γ, respectively.

end if

end for

Now, some terminologies appeared are informally explained in the following.

Affine motion model restricts the motion of each point to depend linearly on its location (*i.e.*, a linear model) plus a constant offset (*i.e.*, a translation) [171].

Motion model verification uses the motion model to validate the consistency of feature pairs (left point, right point). Conversely, feature matches are used to **solve for the model parameters**. All the parameters can be used to form a column vector $\mathbf{x}$, Correspondingly, locations of the left and right points can be encoded in a matrix $\mathbf{A}$ and a column vector $\mathbf{b}$, respectively.

Then, solving for the model parameters $\mathbf{x}$ becomes fitting the point pairs $\mathbf{A}$ and $\mathbf{b}$ to the parameterized model $\mathbf{Ax} = \mathbf{b}$. The solution is given by minimizing a certain error metric such as the sum of square residues: $\mathbf{x} = \arg\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|_2^2$. Since an affine model can be characterized using 6 parameters, the linear system is over-determined as long as there are over 6 point pairs. Although the exact solution may not exist, it is unique if existing. Moreover, a closed-form approximate solution can always be given by Least Squares: $\mathbf{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b}$. However, the linear system can be under-determined in the case with a few point pairs and a complicated model with many parameters. Then, $\mathbf{x}$ can be seen as a dimension-augmented representation of

b. If it is insisted to apply Least Squares, augmented dimension will be hallucinated. In this case, a **sparse** usage of $\mathbf{A}$ can be sought by adding a constraint $\|\mathbf{x}\|_1 \leq T$, which is relaxed from $\|\mathbf{x}\|_0 \leq S$.

Coordinate transformation from image coordinates' to World's coordinates follows the standard geometric model of image formation [171] as shown below.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & s_\theta & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ l \end{bmatrix}$$

In the r.h.s, the first matrices encodes the scaling information and the second encodes the focus. The product of these two matrices is termed the intrinsic parameter matrix. Here the projection matrix is not included since the original coordinates are uncalibrated 2D image coordinates, instead of World's 3D coordinates. However, only linear distortion is considered in this equation. The **radial distortion** has initially first compensated by image rectification [171].

Essential matrix E is a 3×3 matrix encoded in the epipolar constraint and be decomposed to the relative pose/motion $[\mathbf{R}, \mathbf{t}]$ based on the SVD of $\mathbf{E}$ [171].

Quarternion of a rotation. Unit quarternions [185] is a four-element vector. A rotation matrix can be represented by a quarternion as:

$$\boldsymbol{R} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 + q_2^2 - q_1^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_1) & q_0^2 + q_3^2 - q_1^2 - q_2^2 \end{bmatrix}$$

Euler angles of a rotation [186] are computed according to Euler's rotation theorem [188], which implies that the composition $\Delta\mathbf{R}$ of two rotations $\mathbf{R}_{mean}$ and $\mathbf{R}^{-1}$ is also a rotation. From [188], suppose an axis of rotation is specified by a unit vector $[x, y, z]$ and there is an infinitely small rotation of angle $\Delta\theta$ about the vector. Expanding the rotation matrix as an infinite addition, and taking the first-order Taylor series expansion, the rotation matrix $\Delta\mathbf{R}$ is represented as

$$\Delta R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & z & -y \\ -z & 0 & x \\ y & -x & 0 \end{bmatrix} \Delta\theta = \mathbf{I} + \mathbf{A}\,\Delta\theta.$$

Note that $\Delta\mathbf{R}$ is a skew symmetric matrix, where the element x, y, z denotes the Euler angle in X, Y, Z axis, respectively. In Algorithm 5.1, x, y, z correspond to α, β, γ. Namely, α, β, γ are the rotation angle in X, Y, Z axis, respectively.

Re-projection error is a geometric error, which is the distance between a projected point and a measured one [173]. The model projects the held-out query key point using the estimated $[\mathbf{R}, \mathbf{t}]$ and computes its Euclidean distance to its pair.

5.1.3 Experiments

This section presents the experimental results of the feature matches' accuracy, the estimated motion's accuracy, and the estimated motion's sensitivity with respect to the number of motion model used in feature matching.

Implementation. A couple of libraries are used to perform the experiemnt.

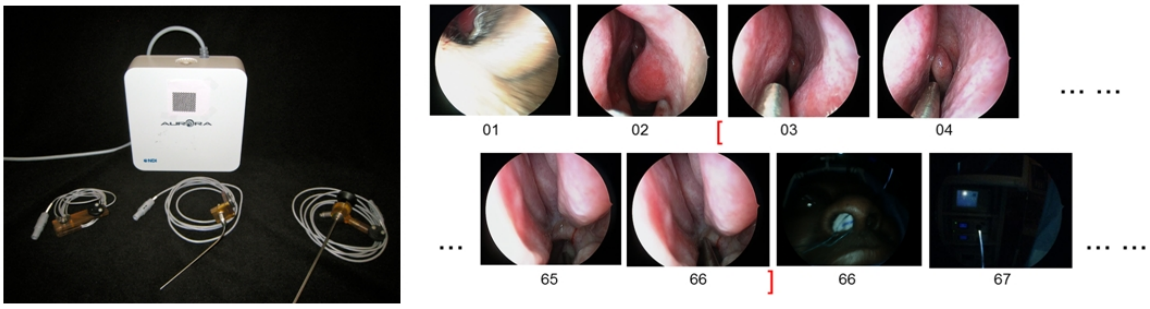


Figure 5.8: The left figure shows the endoscopic sensor and data collection devices. The top box is the processor from NDI. The bottom left is a high-precision optically tracked endoscope, The bottom middle and right form a EM tracked scope for use in airway data collection. The right figure shows an endoscopic video sample of a patient's sinus.

Camera calibration is performed by using Caltech calibration toolkit [189].

SIFT features are extracted using VLfeat library [190].

HMA matching strategy is performed using HMA toolbox [191].

Camera motion recovery from **E** is via Structure and Motion toolkit [192].

Dataset. Fig. 5.8 shows the data collection system developed to simultaneously capture both the endoscopic video and EM tracking data, though EM data are not examined in this dissertation. The collected video lasts for hours at 30 FPS. As summarized in Fig. 5.8, the sequence is down-sampled to be at 1 FPS and select 64 continuous frames, among which 46 frames are with over 4 SIFT key points detected. The baseline between lens in two monocular adjacent views is relatively small. Interested readers may refer to [170] for how sensitive is the estimated motion affected by the baseline. Endoscopic images normally contain scaling and rotation, changes in illumination and 3D camera viewpoint, and low-textured non-planar surfaces. Usu-

ally a few feature key points are detected. Then, the hope lies in a large inlier ratio of matched key point pairs. In certain frames such as frame 03 and 04, the partial occlusion is mainly from the tools and specularities are mostly a result of air bubbles forming on local wet surfaces.

Accuracy of Feature Matching. Between HMA and SIFT, Fig. 5.6 presents a qualitative comparison of matched features. SIFT presents a number of line crossing which implies mismatches. For instance, a SIFT key point in the left region can be matched to another in the right region, which is unlikely to be given by HMA. This is verified in Fig. 5.9 that presents a quantitative comparison of the detected outlier number given by RANSAC verification vs. the total matched feature number given by the tentative matching. It can be seen that although HMA and SIFT generate a similar number of matched features, HMA induces a higher inlier ratio than SIFT does.

Accuracy of Estimated Motion. Now, the held-out query pair is projected using the estimated $[\mathbf{R}, \mathbf{t}]$. For each pair of adjacent frames, the inputs are locations of key points in the frame τ and those in the frame $\tau + 1$, together with the essential matrix $\mathbf{E}$. The Mean Square Error (MSE) for HMA and SIFT are presented in Fig. 5.10 and Fig. 5.11, respectively. In both cases, most frames' MSE are within 5 $pixel^2$, which verifies both HMA and SIFT matches are generally accurate. The next question is how sensitive is the estimated motion affected by the feature matching.

Sensitivity of Estimated Motion. Now, the estimated rotation $\mathbf{R}$ will be exam-

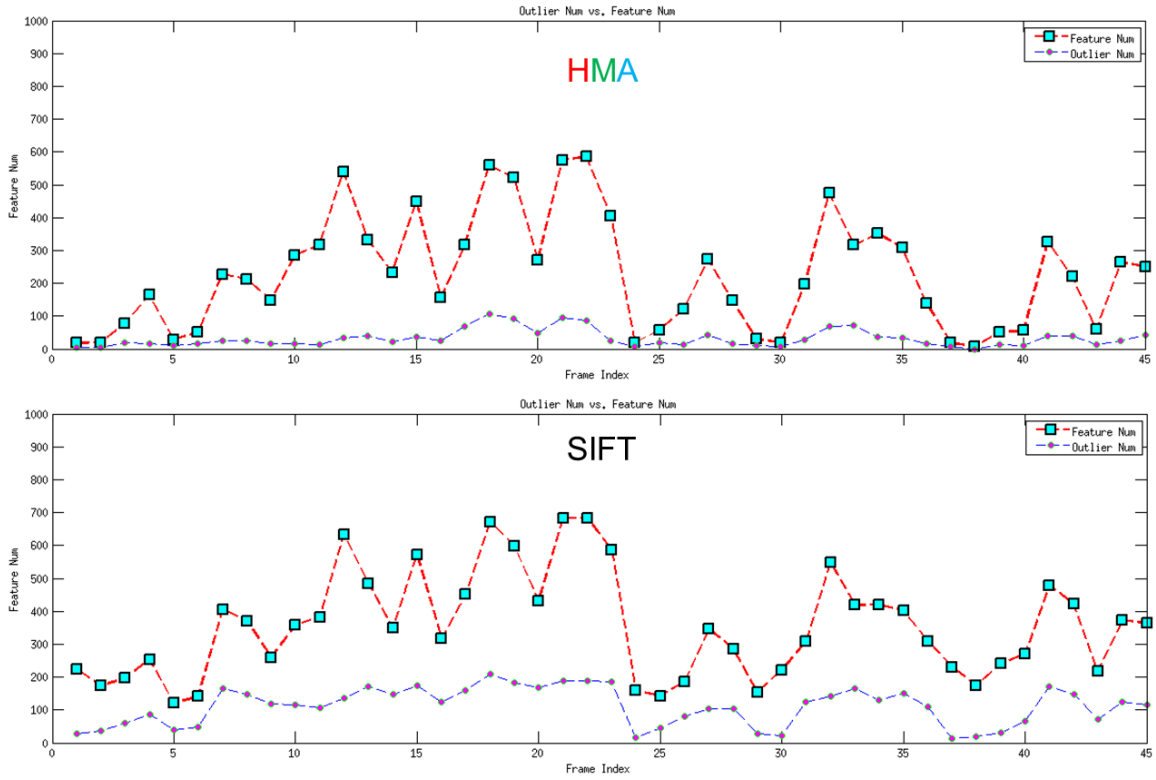


Figure 5.9: RANSAC detected **outlier** number vs. Total matches number. HMA generates much fewer (< 100) outliers than SIFT does (> 100). Better seen on the computer.

ined. The variances of the rotation angle α, β, γ characterize the sensitivity of the rotation. Please review Algorithm 1 for computing the rotation angles. Fig. 5.12 displays the respective standard deviation together with the number of features for both HMA and SIFT. Over time the standard deviation curves of rotation angles follow similar trends, which are generally in the opposite direction of the feature number curve. Namely, when there are more feature matches, the rotation angles are less variant. Notably, the difference of feature number is important only when the features are relatively a few, which is more or less the case in endoscopic images.

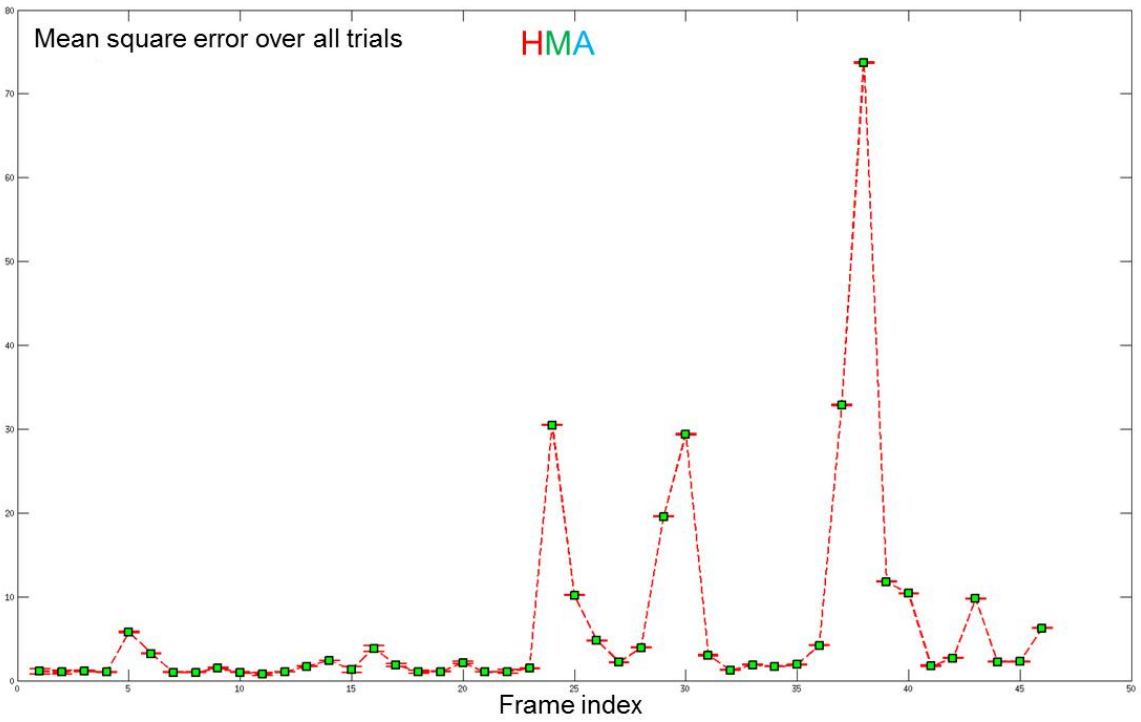


Figure 5.10: Re-projection error (pixel) of the held-out query key point with HMA matching.

Moreover, the number of well-matched features rely on the matching algorithm, Thus, for low-textured scenes such as sinuses with textureless surfaces and specularities, It can be concluded that the estimated motion is sensitive to the number of motion models employed in feature matching.

5.1.4 Summary

This section is about answering three questions. Firstly, does multi-model induce more accurate feature matches than single-model? Secondly, is the camera motion estimated from feature matches given by multi-model more accurate than those given

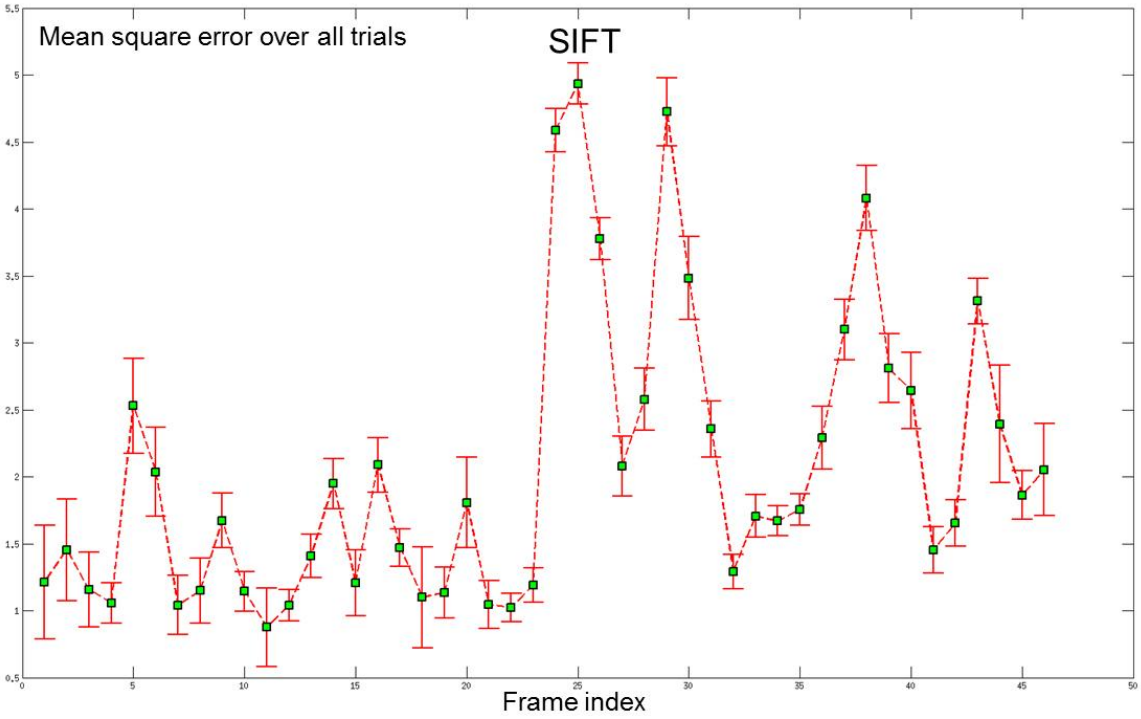


Figure 5.11: Re-projection error (pixel) of the held-out query key point with SIFT matching.

by single-model? Thirdly, how sensitive is the estimated motion affected by the motion model number in feature matching? While the theoretical property of multi-modal feature matching is analyzed in the case of multi-modal RANSAC, It is fully awared that the subsequent motion estimation is of empirical nature in practice. As a result, the above three questions is empirically investigated by conducting cross validation on endoscopic videos of sinuses. It is found that although multi-model (HMA) and a single-model (the original SIFT) generate a similar number of matched features, HMA induces a higher inlier ratio than SIFT does. Besides, both HMA and SIFT matches are generally accurate, for most frames' MSE are within 5 $pixel^2$ in

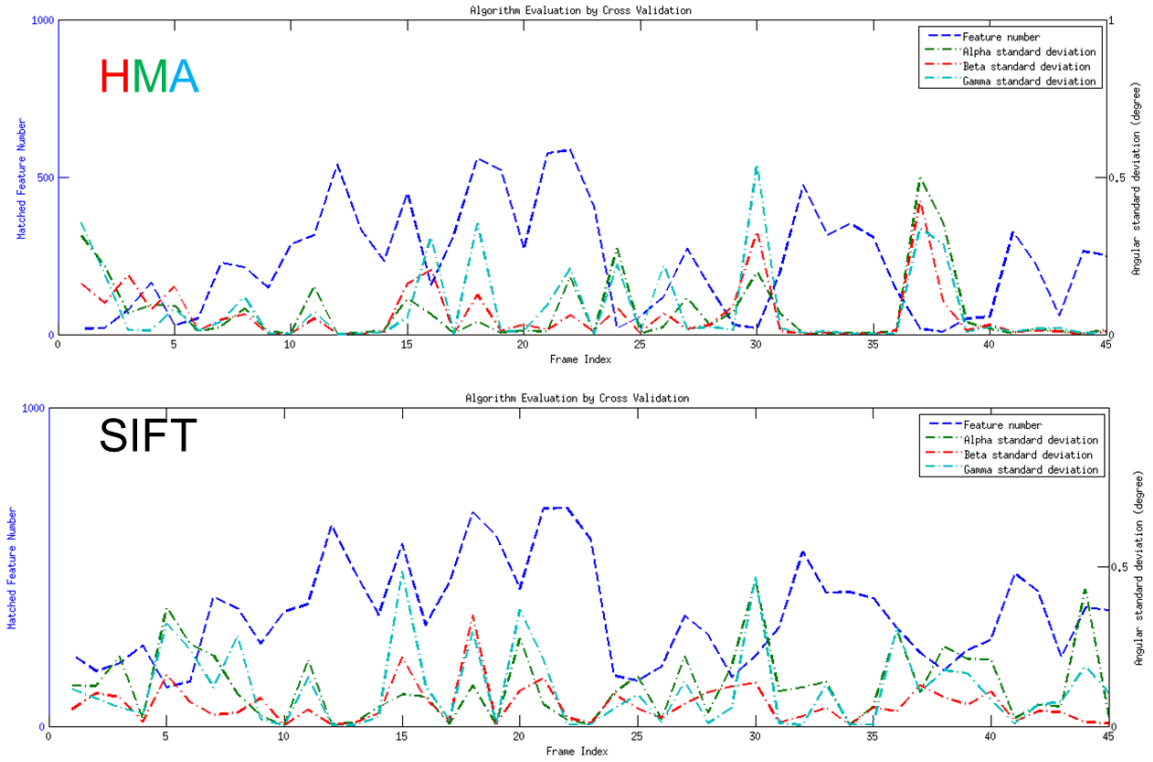


Figure 5.12: The standard deviation of α, β, γ vs. feature number. The left Y-axis denotes the matched feature number, which is displayed in blue. The right Y-axis denotes the angular standard deviation. Better to be seen on computer.

both cases. Moreover, when there are more feature matches, the rotation angles are less variant. It verifies that the estimated motion in low-textured endoscopic scenarios is sensitive to the number of motion model used in feature matching.

5.2 Local motion estimation: action detection with motion model added

Localizing an object is crucial to the success of fine-grained video analysis, so it is desired to design a robust tracker. Online boosting (OB) trackers [193–195] produce competitive performances. OB [193] selects samples and trains a classifier online, making the tracker adapt well to the target's appearance changes. However, the classifier may suffer from sub-optimal samples. Then, online Multiple Instance Learning tracker (MILTrack) [194] is proposed. It bags positive samples and handles the sample ambiguity using MIL.

The camera can move or rotate with changing views (*e.g.*, in aerial surveillance, robot vision, cars surrounding area surveillance). Therefore, the challenges may come from four aspects: 1) variance in scenes (*e.g.*, illumination, occlusion, camouflage distraction, background clutter, *etc.*); 2) variance in the targets appearance (*e.g.*, scale, pose, deformation for non-rigid or articulated objects, *etc.*); 3) not good image quality (*e.g.*, low resolution, low frame rate, intermittent frame corruption, etc.); 4) online processing: incoming stream is tracked in real time, so there is no access to future frames. Nevertheless, it id still desired to succeed at persistent long-term tracking. Top-down cues (*e.g.*, attention [196, 197], trajectory saliency [198]) may be helpful, while this section is more about tracking with appearance-based features. However, it is open question regarding how to make the appearance-based tracker

robust and not drift away.

Online adaptation of appearance model is a double-edged sword. If the predicted target location is not accurate, the appearance model will get updated with a sub-optimal positive example. Over time this can degrade the appearance model and can cause drift. An extreme example is a scenario where the target completely leaves the field of view. The online classifier gets updated with background samples. Then, when the target re-appear, the classifier is unlikely to discriminate it from the background. It is also found that most of the online adaptation methods adopt online boosting. Note that off-line boosting adopts the strategy of batch gradient descent, namely it needs the entire training data at once. In contrast, the online variant proposed by Oza [199] uses stochastic gradient descent, which assumes that the loss over the entire training data can be expressed as a sum of the loss for each point.

5.2.1 Related works: multiple instance learning (MIL) boosting

In [194], Babenko *et al.* do not avoid the ambiguity of sample selection as done in the semi-supervised boosting. Instead, they use Multiple Instance Learning (MIL) [200, 201] to bag positive samples. Then, the ambiguity is passed to the learning algorithm rather than the classifiers input. They integrate MIL into the online boosting framework to train a strong classifier based on selected weak clas-

Figure 5.13: Performance comparison of typical trackers. From left to right: frame #0, #50, #100, #150, #200.

sifiers. Their strategy for searching the most likely location is a greedy manner by simply taking the location with the maximum classifier response. If SVM is used instead of boosting to train the classifier, the learning problem can be solved by latent SVM [202]. In addition, Grabner *et al.* perform it in a semi-supervised manner [203], which is designed specifically to address the drift problem. Here, the labeled data only comes from the initial frame, and samples from subsequent frames are all used as unlabeled data. The basic idea is to combine an off-line trained prior classifier and online trained classifier. The combined classifier cannot deviate too much from the off-line classifier. This limits drifting, though restricting the adaptation too much and throw a lot of useful information.

Fig. 5.13 compares the performance of a couple of tracking methods, the typical ones of which are online boosting-based tracking by detection methods. The original videos frame rate is 30 frm/s, with image size of 640 $\times$ 320 (pixel). # 100: the target person is occluded by another person; # 150 200: the left person is the target. (a) Online boosting [16][17] with fixed scale and features of Haar-like, HOG and LBP. (b) Semi-supervised online boosting [19] with fixed scale and features of Haar-like, HOG and LBP. (c) MIL-Track [22] with fixed scale and Haar-like feature. (d) MILTrack with fixed scale and HOG feature. (e) TLD tracker [25] with adaptive scale and a simply designed feature. It tracks the target as a circle. (f) Basic template matching [26]. (g) Basic Mean Shift [31]. (h) Frag-Track [32]. (i) KLT optical flow [41]. Each corner point is drawn with an arrow (motion vector). (j) Particle filter [44], with

all samples drawn. (k) Frame-by-frame detection results of state-of-theart human detector - Deformable Part Model [23].

5.2.2 Related works: Bayesian filtering

In terms of particle filtering, z_t and x_t are used to represent the targets motion state and appearance (*e.g.*, positive or negative sample) at time t. Further, the model uses $z_{1:t}$ and $x_{1:t}$ x to denote the state sequence $\{z_1, ..., z_t\}$ and the appearance sequence $\{x_1, ..., x_t\}$. To simplify the calculation of those posterior probabilities, two hypotheses of conditional independence are proposed: a) Markov property of the motion transition model $p(x_t \| x_{t-1})$. b) Independence of the appearance observation Model $p(z_t|x_t)$. Suppose: initially, the prior distribution $p(z_t|x_t)$. Then $p(x_t|z_{1:t})$ can be obtained through two-step iteration of estimation and update.

Algorithm 5: Bayesian filtering.

Input: $p(x_{t-1}|z_{1:t-1})$ with the transition model $p(x_j|x_{j-1})$ and observation model $p(z_j|x_j)$, where $j = 1, ..., t$.

Output: $p(x_t|z_{1:t})$, namely x_t.

1) Prediction. At time $(t-1)$, the prior probability $p(x_t|z_{1:t-1})$ is predicted at time t.

2) Update. At time t, the appearance z_t is extracted and the motion state's posterior probability $p(x_t|z_{1:t})$ is computed.

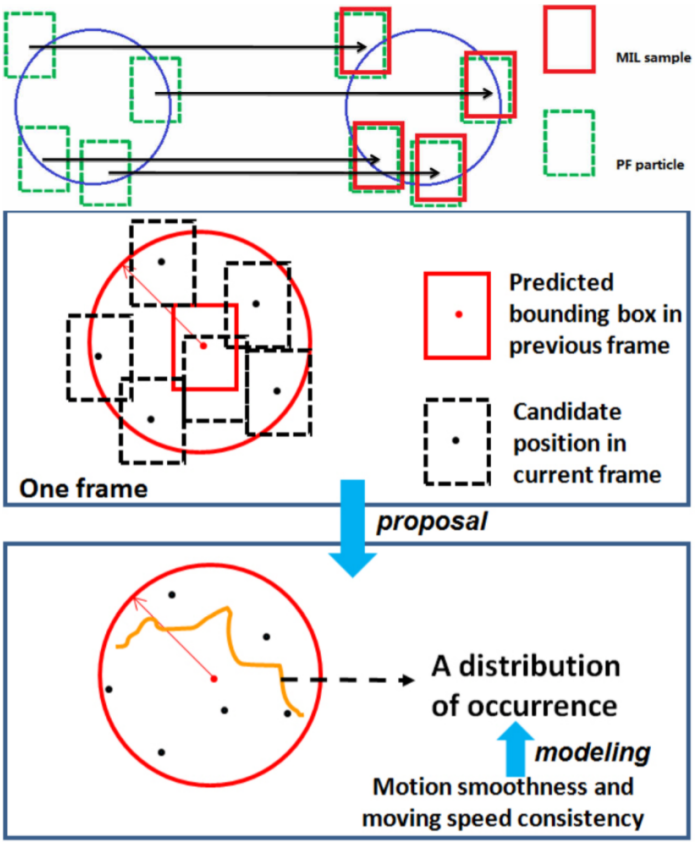


Figure 5.14: Relationship between the MIL tracker and particle filter tracker.

5.2.3 MIL boosting tracker with particle filter as motion model

It has shown promising results, while it assumes a uniform distribution of the candidate locations within the search region and directly chooses the location with the maximum response. That makes the results lack of motion smoothness and moving speed consistency. As shown in Fig. 5.14, it is proposed to model the occurrence distribution according to the motion smoothness and moving speed consistency [204].

The Particle Filters (PF) is incorporated [204] into MILTrack by setting each particle as a sample for MILTrack. The strengthened motion model induces robustness to distraction and occlusion. The *Weighted Response* of each particle is defined as

$$Weighted\ Response = Response * Occurrence\ Probability,$$

where *Response* is the classifiers' output, *Occurrence Probability* represents the similarity (*i.e.*, *Likelihood*) between the particles and the initial image patch of the target. In PF, the motion state X (*i.e.*, location, scale) is sampled from a Gaussian window around the previous state and predicted using a 2nd-order auto-regressive dynamical model [204] as $X_t - \overline{X} = A_2(X_{t-2} - \overline{X}) + A_1(X_{t-1} - \overline{X}) + B_0 w_t$, where $\overline{X}$ denotes the previous particles' mean, the subscript denotes frame number, w denotes prediction noise, A_1, A_2, B_0 are coefficients, and $A_1 + A_2 = 1$. This algorithm is named *MIL-PF*, which is explained below.

Algorithm 6: MIL-PF. Performing the following frame by frame.

(1) PF: for each particle, apply the transition model to predict the position and scale, and the observation model to estimate the *Likelihood*.

(2) Use the position and scale parameters of each particle to generate a corresponding sample for MILTrack.

(3) MILTrack: for each sample, employ the classifier to calculate the *Response*.

(4) Normalize *Likelihoods* and *Responses*.

(5) For each sample, calculating: *Weighted Response = Response * Likelihood*.

(6) Select the sample with max *Weighted Response* and update the state.

5.2.4 Experiments

MIL-PF is implemented based on codes from [194] and [204]. It runs at about 10 fps (360× 240) on a PC with Intel Core 2 Duo CPU. It is compared with OB [193], MILTrack [194], FragTrack [205] and TLD [195]. Their released codes are executed with the same initialization. OB, FragTrack and MILTrack do not search in scale space, while TLD and MIL-PF both do. For MILTrack and MIL-PF, all the common parameters are kept the same and fixed. Images are represented by Haar-like features for MILTrack and color histograms for PF. For MIL-PF's transition model, A_1 is 2.0, A_2 is -1.0, B_0 is 1.0, and w is generated from a Gaussian distribution randomly.

The trackers are evaluated on the Youtube dataset [1] with 50 sequences. The ground truths are labeled for the bounding box and the silhouette. Evaluation results for multiple trials are averaged. First, the very challenging 48th sequence will be discussed. Fig. 5.19 plots the center location error versus the frame number. Fig. 5.20 plots the overlap score versus the frame number. The overlap score is defined as the fraction of the intersection to the union of ground truth and tracking result. As a convention, a hit means that the score is greater than 1/3 (the black horizontal line). It is found that MIL-PF achieves the best overall performance, and is especially better than MILTrack. OB's performance is poor, possibly suffering from suboptimal and gradually imprecise samples. Over time they degrade the classifier's discriminative power and cause drifts from the target. Thus, MILTrack performs better than OB. However, from frame 40, both MILTrack and OB suddenly perform worse, possibly

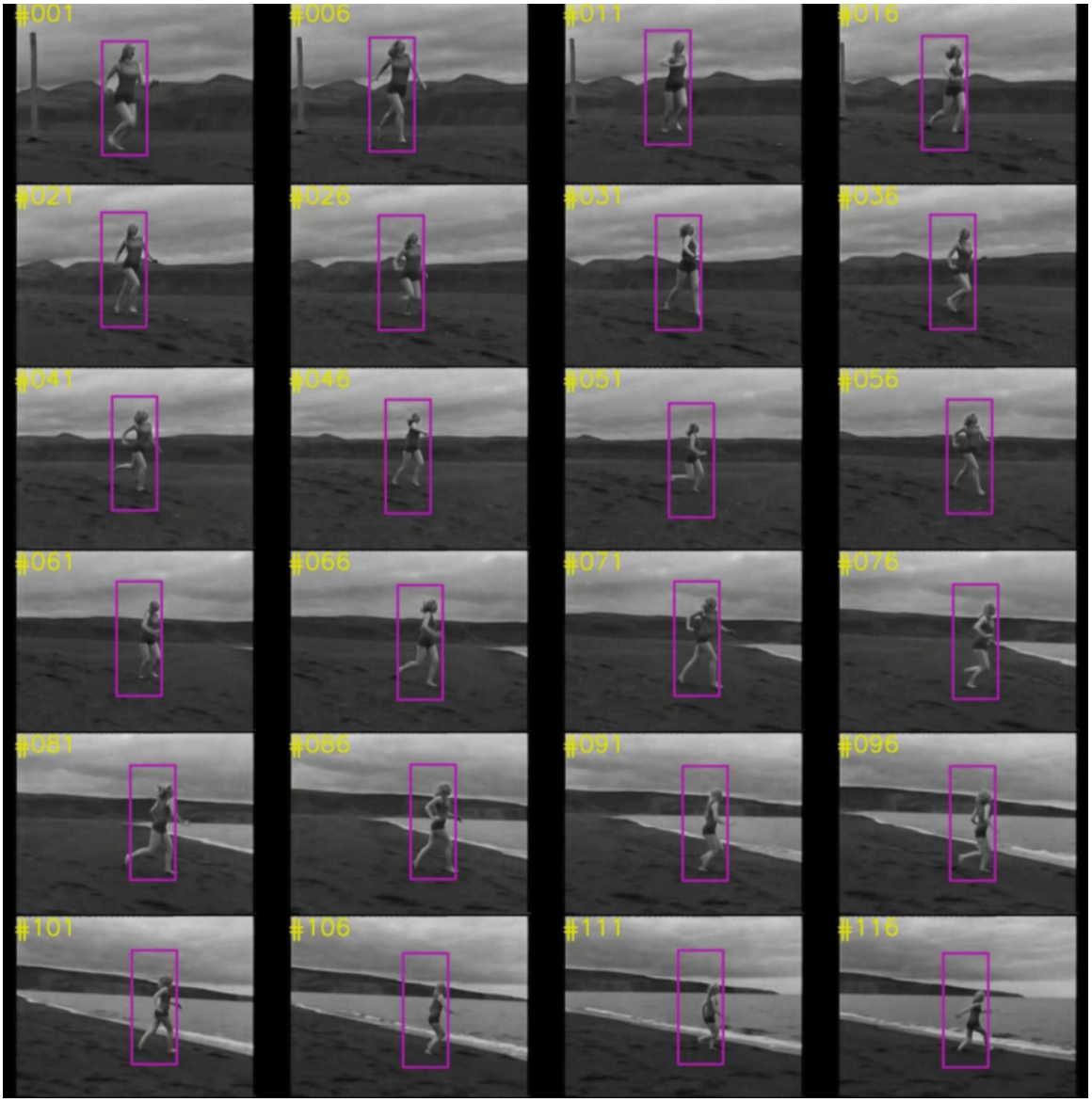


Figure 5.15: Example tracking results given by the proposed MIL-PF on the 5th sequence of the Youtube dataset. The frame index increases from left to right and then per row from top to bottom row by row.

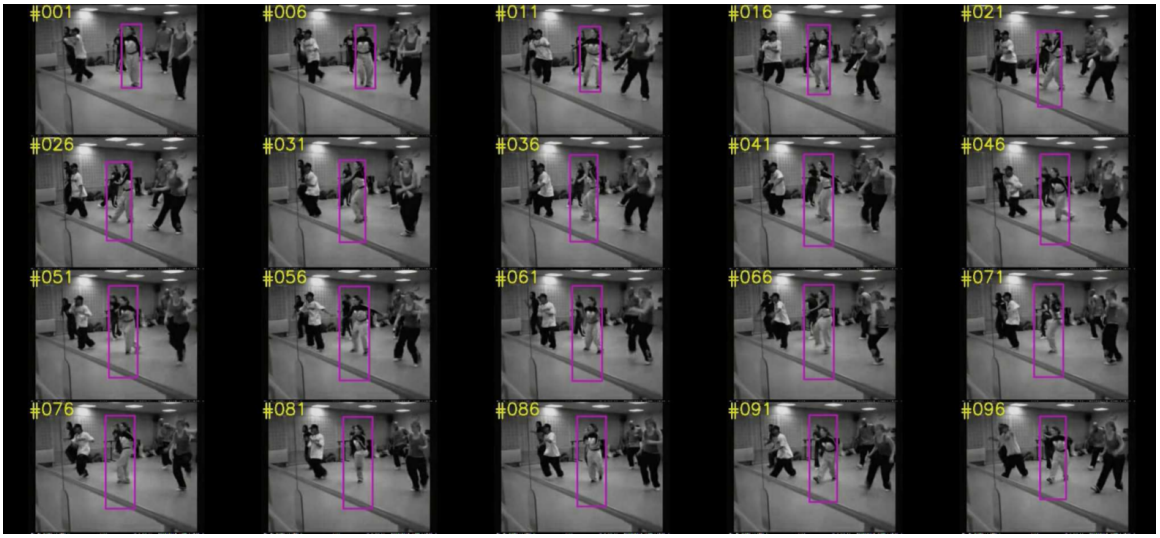


Figure 5.16: Example tracking results given by the proposed MIL-PF on the 7th sequence of the Youtube dataset.

due to the lack of motion model for occlusion handling.

Fig. 5.21 shows the per-video overall performance by plotting the percentage of frames in which the estimated target location is within a threshold distance of the ground truth. It plots the precision versus threshold. Here the Precision is defined as the percentage of frames that have the predicted center location with the Threshold number of pixels off from the ground-truth center location. Here the Precision is counting the frames, not computing the intersection/union per frame. A more accurate name is the Ratio of Successfully Frames. In this defination, the Recall is (1−Precision) The definition of success is the distance between ground truth and prediction in pixels. Basically, moving towards left requires a more accurate tracker.

The precision is compared at threshold 30 in the curve, to find that MIL-PF achieves the highest precision. Below 30, MIL-PF and TLD closely lead the perfor-

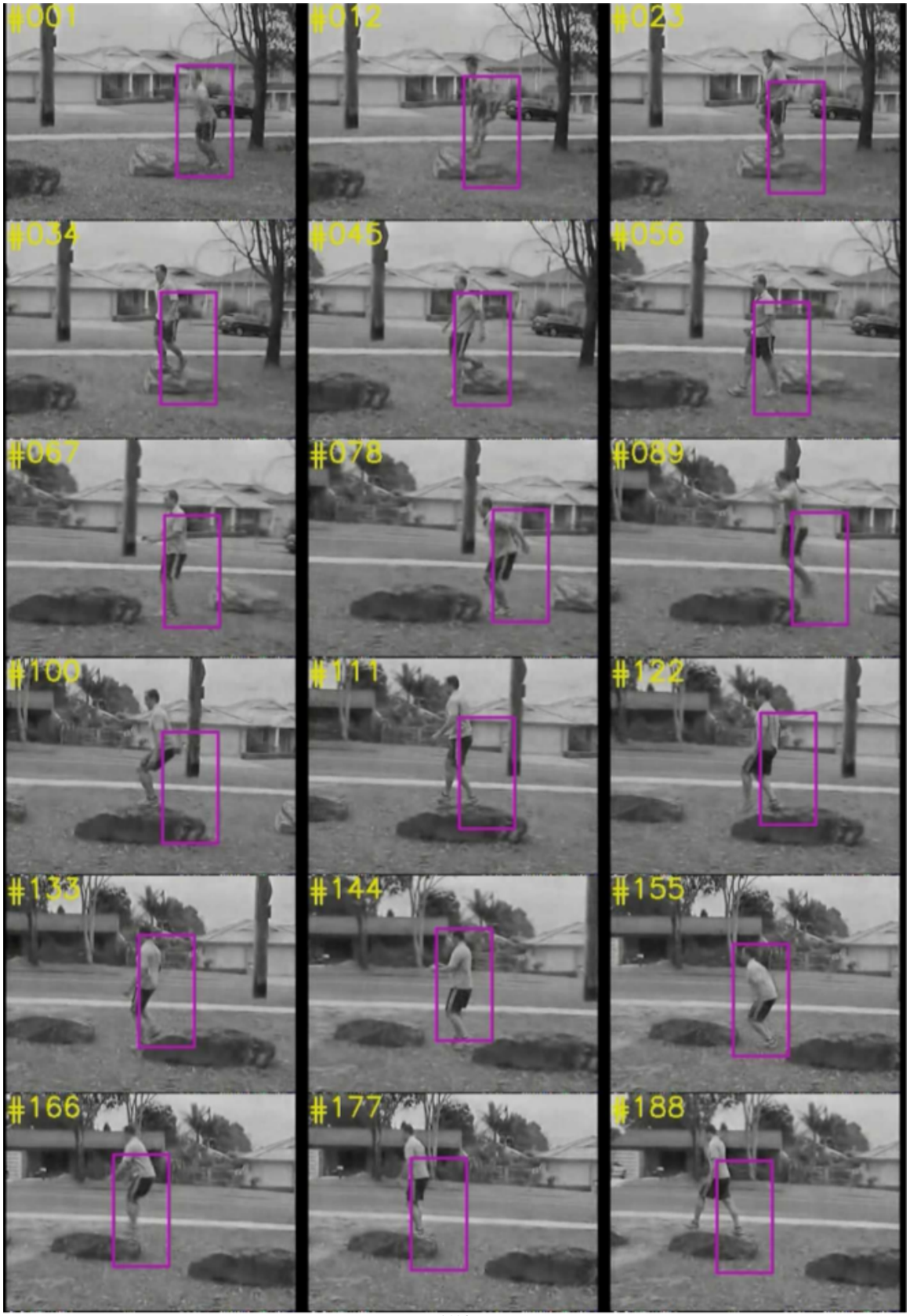


Figure 5.17: Example tracking results given by the proposed MIL-PF on the 38th sequence of the Youtube dataset.

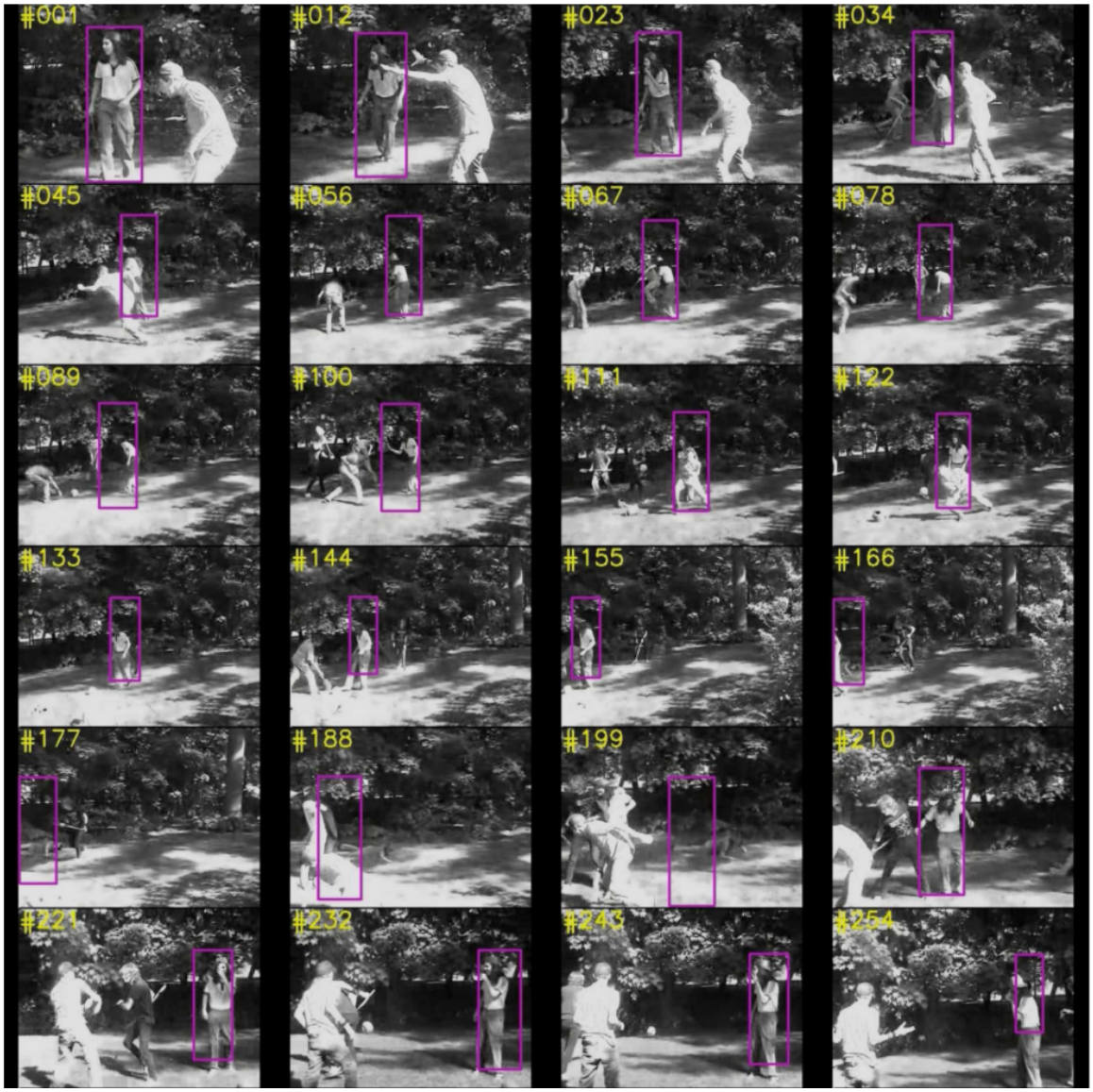


Figure 5.18: Example tracking results given by the proposed MIL-PF on the 48th sequence of the Youtube dataset.

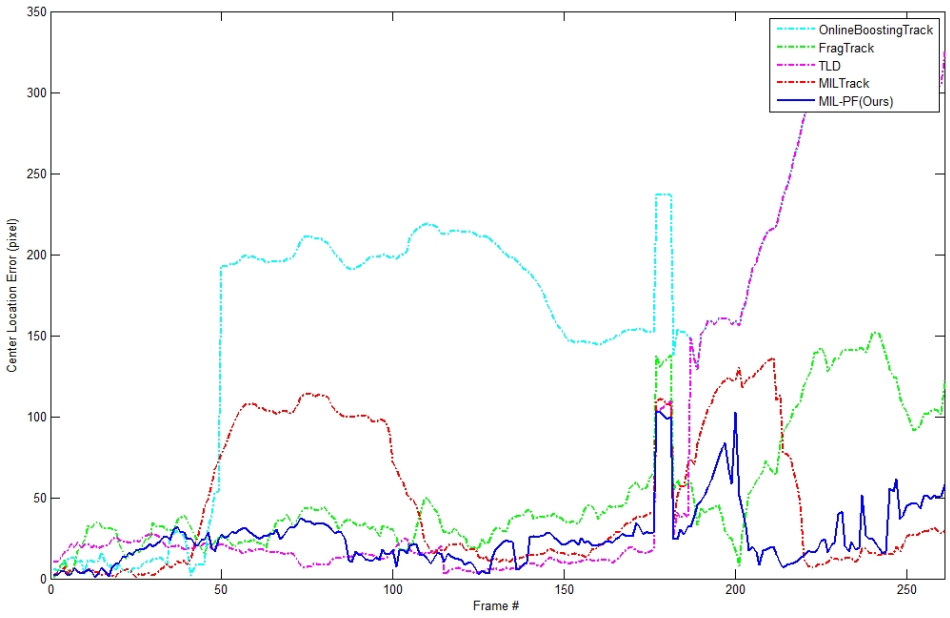


Figure 5.19: Tracking result evaluation for seq48 [1]. Center location error plots (in color). See text for details.

mance; above 30, MIL-PF has a big lead. For the whole dataset, Table 5.1 summarizes the average Precision over all video in the YouTube dataset at threshold 30 pixels. MIL-PF achieves the best overall performance. The results are not only stable but also generally accurate.

Method	OB	FragTrack	TLD	MILTrack	**MIL-PF**
Ave. Prec.	0.41	0.56	0.77	0.62	**0.83**

Table 5.1: The precision (Ratio of Successfully Frames) at threshold 30 on Youtube dataset [1].

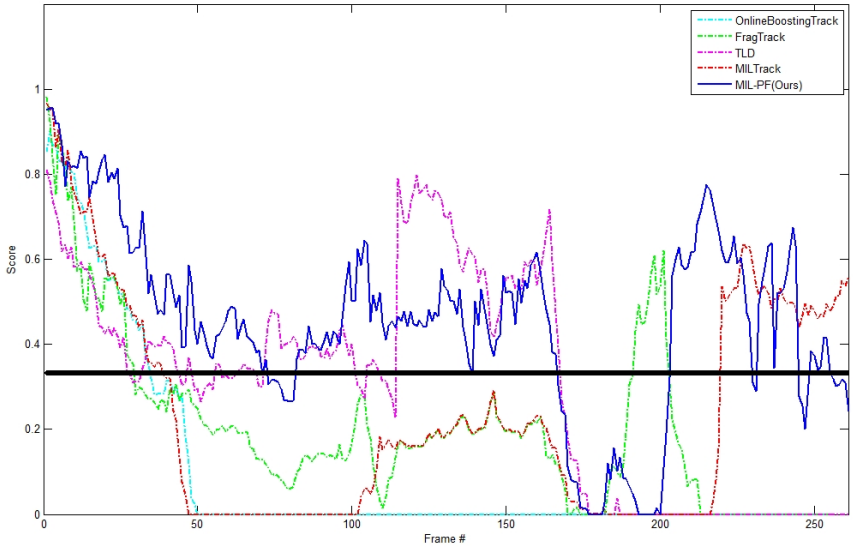


Figure 5.20: Tracking result evaluation for seq48 [1]. Overlap score plots (in color). See text for details.

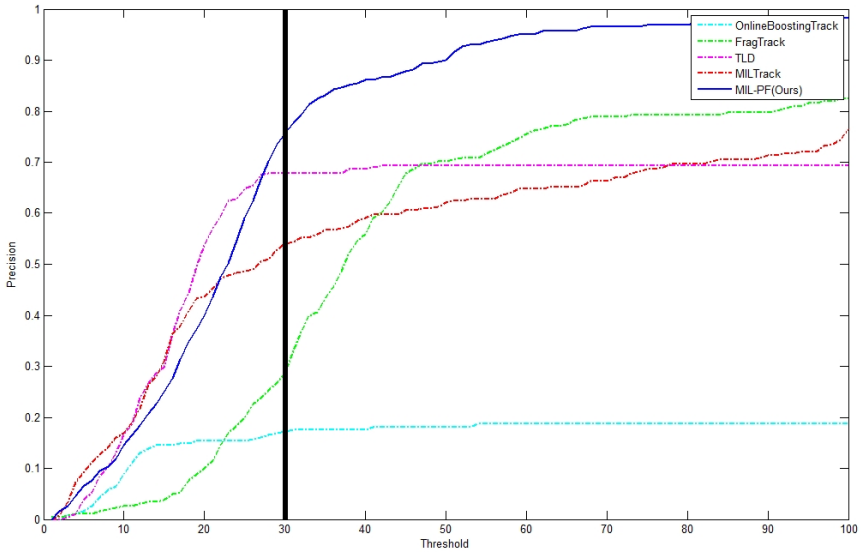


Figure 5.21: Precision plots (in color). See text for details.

5.2.5 Summary

A robust tracker is designed called MIL-PF combining the merits of the Multiple Instance Learning tracker (MIL tracker) and Particle Filter (PF). MIL tracker optimizes the selection of samples for learning an appearance model, while PF strengthens the motion model of MIL tracker. Experimental results verify that this fusion induces a robust tracker providing tracking results with high precision.

MIL-PF is mainly based on tracking with online multiple instance learning (MIL-Track), but with response weighted by Particle Filtering (PF). Boosting-based MIL-Track can train a strong classifier to discriminate target from the background based on appearances, while the introduction of PF strengthens its weak motion model.

It is not suggested to take MIL-PF as the state-of-the-art although it does achieve the state-of-the-art performance on the Youtube Dataset under the Average Precision evaluation measure by the time it was published. The Average Precision is essentially averaging the rotios of successfully tracked frames per video. It is not the standard Precision in the setting of Precision-Recall. However, the take-way message is that bringing in a motion model to tracking-by-detection models is beneficial. Given the rapid progress on the deep learning based object detection, tracking-by-detection models will continue to dominate the tracking area, while tranditional model components can be generalized to them as well to give further performance boost.

5.3 Conclusion

In this chapter, the first example work investigates how sensitive the estimated motion is affected by the number of motion models assumed in feature matching. While the sensitivity can be analytically evaluated, an empirical analysis is presented in a leaving-one-out cross validation setting without requiring labels of ground-truth matches. Then, the sensitivity is characterized by the variance of a sequence of motion estimates. A series of quantitative comparisons are presented such as accuracy and variance between Multi-Affine motion models and a global affine model used in SIFT. This work has been published in our paper [206].

The second example work in this chapter is about spatially action localization or called object tracking. A robust tracker called MIL-PF is designed by combining the merits of the Multiple Instance Learning tracker and Particle Filter tracker. It has been shown that weakly supervised online manner does not necessarily result in better robustness, but does enable recovering from the error. In addition, it is a good choice to maintain an off-line classifier together with the online classifier. Moreover, Particle filter tracker cannot track the target very accurately, but does not drift far away. However, once combined with the state-of-the-art tracking by detection methods, the motion estimator given by particle filtering is informative and supplemental to appearance-based tracker. This work has been published in our paper [10].

Chapter 6

Spatiotemporal Representation of Action Videos as 3D Graphs

There is a huge amount of how-to videos online such as WikiHow channels at YouTube. Nowadays when a person does not know how to do something, he or she normally turns to the Internet and very likely an instructional video. Nowadays, computers are expected to be able to not only assess skills of specific training tasks but also assess relative skills in how-to videos by ranking videos based on the skill.

Firstly, it is aimed to design an object-class independent framework to segment moving objects in videos with dynamic scenes. Although the experiments are mostly shown for human, it is aimed to design a generic model that is independent of object class (say, car, plane, animals). Our intuition is to track the object by methods developed in the previous chapter and then segment it locally, which in turn makes

it amenable to accurately assigning seed labels and applying priors learned for the object. Moreover, the rich labeled data on the Web makes such prior learning feasible. In detail, an online Web-data-driven framework is presented for segmenting moving objects in videos. This framework uses object shape priors learned in an online fashion from relevant labeled images ranked in a large-scale Web image set. The method for online prior learning has three steps:

(1) relevant silhouette images for training are online selected using a user-provided bounding box and an object class annotation;

(2) image patches containing the annotated object for testing are obtained via an online trained tracker;

(3) a holistic shape energy term is learned for the object, while the object and background seed labels are propagated between frames.

Afterward, the segmentation is optimized via 3D Graph cuts with the shape term and soft seed assignments. The performance is evaluated on the challenging Youtube dataset and found to be competitive with the state-of-the-art that requires offline modeling based on pre-selected templates and a pre-trained person detector. Comparison experiments have verified that tracking and seed label propagation both induce less distraction, while the shape prior induces more complete segments.

In the second work, a video is represented as a 3D graph in the deep learning setting - spatiotemporal 3D ConvNet. An action quality regressor named S3D is proposed to fuse segment-level P3Ds on top of ED-TCN using a regression layer.

6.1 Spatial action segmentation using 3D Graph cuts and shape priors

Moving object segmentation is useful yet challenging in video analysis, such as action recognition and person identification. Particular challenges come from three aspects. First, most videos include dynamic scenes, *e.g.*, moving background and global motion. Second, it is preferred yet difficult to segment an arbitrary object automatically with less user intervention [207–209]. Third, efficiency is also important due to the volume of video data. To address those, an object-class independent framework is proposed to segment selected moving objects using priors driven by large-scale Web data. It requires only minimal user input and no pre-trained detectors for respective object classes. As shown in Fig. 6.1, the novelty lies in the paradigm of learning priors online. Since dynamic scenes need to be tackled, our intuition is to track the object and segment it locally, which in turn makes it amenable to applying priors learned for the object. Moreover, the rich labeled data on the Web makes such prior learning feasible. *e.g.*, [210] uses *LabelMe* [2] to learn regional segments, [211] uses Web images to learn actions, [212] uses Web videos to learn behavior priors, and [213] uses Youtube videos to learn detectors. The contributions of this work include a method for the online learning of priors naturally used in Graph cuts and a robust tracker combining the merits of two up-to-date methods [194] and [204].

In scenes with a stationary background, *background subtraction* performs well.

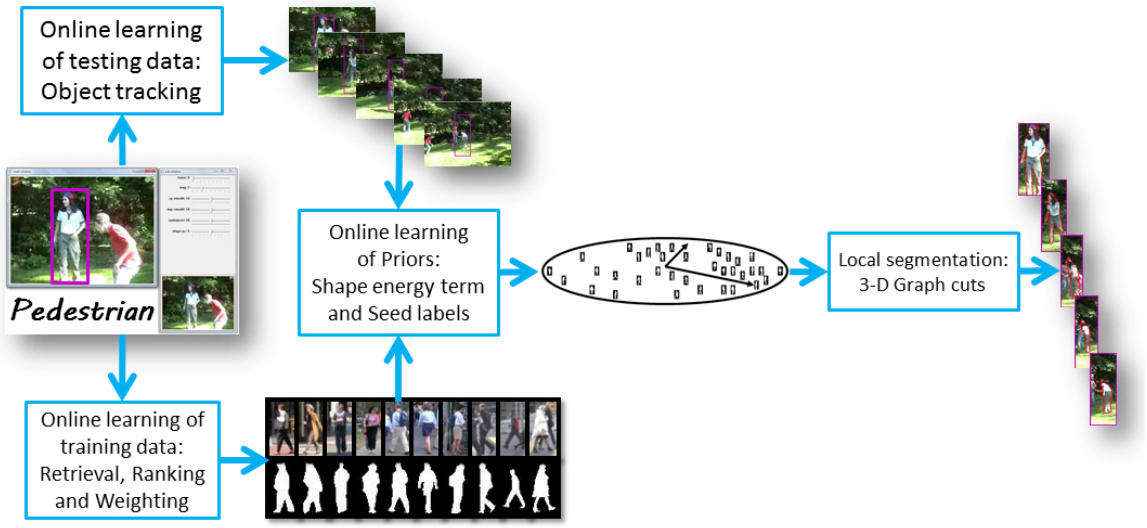


Figure 6.1: Framework overview. First, a user draws a bounding box on the object of interest and provides descriptive keywords. Then, the object is tracked through the video. Meanwhile, a weighted training set is retrieved to learn a shape prior and generate seed labels. Finally, segmentation of the object is produced. Shown for Youtube-seq48 [1].

However, *graph-based segmentation methods* [214–216] often produce better performance because they can incorporate precise constraints and give a better balance of the region and boundary properties [217]. In dynamic scenes, global graph optimization does not work well. One solution is to incorporate object tracking [218, 219]. Simultaneous tracking and segmentation can preserve the temporal coherence [1, 207, 216, 218–221]. For example, Bugeau and Perez use *Mean Shift* for clustering and minimize the energy with MAP/MRF [220]. The descriptor is formed with photometric, spatial and optical flow features to cluster consistent points. Ren and Malik integrate static and temporal cues to segment objects repeatedly [218]. Niebles *et. al.* first use the *pictorial structure* to extract the human volume [1], and then extend that by

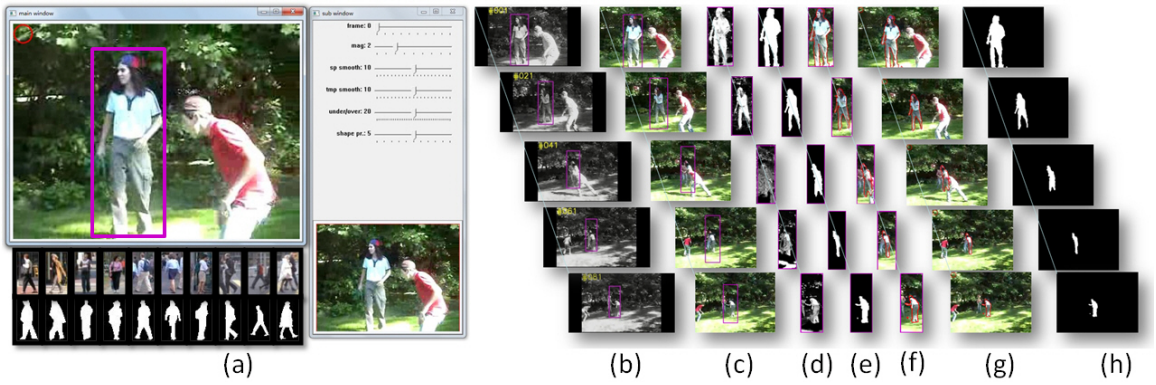


Figure 6.2: A simple interaction enables segmenting a moving person. Youtube-seq48 [1]. (a) Initialization: interaction on UI and sample retrieval (Sec. 6.1.1). (b) Tracking results. (c) Enlarged bounding boxes. (d) Foreground data term (Sec. 6.1.3.3). (e) Silhouettes obtained via 3D Graph cuts (Sec. 6.1.3.3). (f) Contours (*i.e.*, segments). (g) Segments in original images. (h) Silhouettes in large images.

incorporating bottom-up appearances [221].

Another solution is to improve the standard Graph cuts [217]. Modifications mainly differ in the defination of energy terms. Malcolm *et. al.* aim to incorporate a non-uniform prior into the region term, by first penalizing pixels according to their distance from the expected location [207] and then using the *Kernel PCA (KPCA)* pre-image (*i.e.*, reconstructed feature) as a prior occurrence map [222]. [223] embeds a shape probability map into both region and boundary terms. Some works try to design a separate shape term, such as [208, 224]. Recently, it is popular to learn priors such as shapes either from previous segments transductively [208, 223, 225, 226] or from a training set [221, 227]. The training set can be pre-selected for a specific class [221, 227, 228], while an online query is flexible and provides a wide variety and representative samples. For instance, [229] uses KPCA to learn shape priors. *Collect-*

cut [228] iteratively refines segmentation results by clustering seeds and using data-driven top-down cues, which even include a localization prior and object-like regions called key-segments, in their recent extension work [230]. Meanwhile, researchers use *Gaussian Mixture Models (GMM)* [215], *3D histogram* [214] or *Kernel Density Estimation* [226] to assign seed labels. Further, seed label propagation can be achieved by KPCA [222] or self-training [209], *e.g.*, Malcolm *et. al* use the pre-image [231] of a KPCA projection to induce an iteratively refined shape [222].

In short, the standard Graph cuts [217] is improved by embedding a shape term and propagating seed labels. A posterior is used to express the regional term, and present a working system that makes good use of the interactive annotation of the object and the keyword description. The system has three steps.

(1) Online selection of training data from the Web. 1) Given a video, the user draws a bounding box (denoted as **B**) on an object of interest and provides one or more descriptive keywords, *e.g.*, "*pedestrian*"; "*face*"+"*profile*"; "*car*"+"*aerial*"; "*apple*"+"*pad*", *etc.* 2) Inside **B**, a coarse segment **C** is acquired via a fast and automatic variation of *Level set* [232], which uses boundary features (*i.e.*, signed distance function). 3) Keywords are used to retrieve training samples on-the-fly, while the coarse segment is used to select the most similar ones. All training samples are cropped out according to the aspect ratio of the initial bounding box and then scaled to the same size.

(2) Online selection of testing data via object tracking. **B** is tracked through

the video. The returned positions and sizes are generally accurate. To ensure image patches in bounding boxes fully contain the object, the tracking result is enlarged with a margin. The image patch sequence is divided into volumes using a *temporal sliding window*. Even if tracking fails, the system detects it and requests the user to re-initiate tracking. The tracking method has been proposed in the previous chapter.

(3) Segmentation with online learning of priors. The shape energy is learned from the KPCA space. And the object and background seed labels are propagated using the KPCA pre-image of the previous segments. For each volume, 3D Graph cuts are performed with the shape term in the objective and hard constraints from propagated seeds.

In the following, this section first introduces how training data are selected in an online manner. Then, to apply Graph cuts, a discussion is presented on the construction of the objective function and hard constraints. Sec. 6.1.2 aims to learn a holistic shape term from retrieved data. The learning method is expected to extract nonlinear structures and be robust to noises. [231] indicates that KPCA meets such requirements. Sec. 6.1.3 aims to propagate seed labels across frames using KPCA (see also Fig. 6.3).

6.1.1 Online Selection of Training Data

Training samples are retrieved on-the-fly from *LabelMe* [2], a large-scale and growing Web image set. Till now, it contains about 70,000 annotated images, with rich

Keyword	person	pedestrian	human	face	car
Number	13175	651	577	3019	21617
Keyword	plane	motorbike	bike	boat	dog
Number	195	480	1092	1281	250

Table 6.1: Examples of the query result number from *LabelMe* [2].

object categories covered according to *WordNet*. If an image patch of the object is retrieved, its silhouette image with the same size is also extracted. For most classes, the retrievals are abundant, such as "*person*" with 13175 returns, "*car*" with 21617 returns, "*bike*" with 1092 returns, "*boat*" with 1281 returns, *etc.* In comparison, a pre-selected set of 100 templates are used in [221]. Then, the top N representative returns are selected according to the *shape similarity* defined below. For the "*person*" task on a Youtube dataset [1], $N = 500$ is empirically set because the effect on the segmentation plateaus after that. Among the selected samples, the object similarity still can vary. Thus, a weight ω is assigned to each sample. Besides the *shape* similarity, the weighting also considers *color* and *texture*. The readers may temporarily skip the criterion definitions below.

Shape. The sample with more similar object shape with **C** should weight more (the object is termed as foreground). Thus, the shape similarity r_s is measured via the bidirectional *Hausdorff distance* D_h between two contour pixel sets $con_{tst} = \{a_1, ..., a_m\}$ and $con_{trn} = \{b_1, ..., b_n\}$, where *con* denotes a segment's outer contour,

tst denotes the testing video's initial patch, and *trn* denotes a training sample. Next, r_s is defined as $r_s = \frac{1}{D_h(con_{tst}, con_{trn})}$. D_h measures the maximum mismatching degree between two point sets and is defined as

$$D_h = \max\big(d(con_{tst}, con_{trn}), d(con_{trn}, con_{tst})\big),$$

where

$d(con_{tst}, con_{trn}) = \max_{a \in con_{tst}} \min_{b \in con_{trn}} \|a-b\|$ and $d(con_{trn}, con_{tst}) = \max_{b \in con_{trn}} \min_{a \in con_{tst}} \|b-a\|$. $\|\cdot\|$ is the distance between two pixels. The larger r_s is, the more weights the sample possesses. The model ranks r_s decreasingly and selects the top N samples. The corresponding silhouette images form the *training set*.

Color. Statistics show that color values of the foreground (**fg**) and background (**bg**) scatter in different color ranges. Therefore, the contour resulting in more distinct fg and bg distribution should weight more [221]. The color criterion r_c is defined as the χ^2 *distance* between the fg and bg color histograms H_{fg}^{col} and H_{bg}^{col}: $r_c = D_{\chi^2}(H_{fg}^{col}, H_{bg}^{col})$. The fg/bg color distribution is commonly assumed to obey a mixture model (*e.g.*, *GMM* [215]). Besides, D_{χ^2} is defined as $D_{\chi^2}(A, B) = \sum_i \frac{(|A_i|-|B_i|)^2}{|A_i|+|B_i|}$, where $|A_i|$ denotes the number of votes in A's i-th bin.

Texture. A contour locates where the local image pattern changes - from texture blocks to structure blocks. Thus, the sample with more distinct fg-contour local entropy distributions should weight more. This texture (entropy) criterion r_e is defined as the χ^2 *distance* between entropy histograms H_{fg}^{ent} and H_{con}^{ent}: $r_e = D_{\chi^2}(H_{fg}^{ent}, H_{con}^{ent})$. A pixel's local entropy feature is computed within its 8-neighborhoods as $e = -\sum_{i=1}^{i=9} p_i$.

$\ln p_i \sum_{i=1}^{i=9} p_i \cdot (1 - p_i)$, where $p_i = f_i / \sum_{j=1}^{j=9} f_i$ and f is the pixel value. The approximation is based on the *Taylor expansion*.

Weight learning. The three criteria are fused in a *logistic function* by defining the weight as $\omega_i = \frac{1}{1+\exp\left(-(\kappa_0+\sum_{j\in\{s,c,e\}} \kappa_j \cdot r_j)\right)}$ *s.t.* $\sum_i \omega_i = 1$. The coefficients κ_j ($j \in 0, s, c, e$; κ_0 is a constant) are learned from all the N selected exemplars using *logistic regression* with a *predictor* and a *response vector*. The *predictor* records each exemplar's r_j ($j \in s, c, e$), and the *response vector* records if an example is chosen. These criteria generally guarantee the relevance of training data (silhouette images) even when the keyword is not exactly accurate.

6.1.2 Shape Energy Term

Kernel feature space. Suppose K training samples are retrieved from the labeled Web data. Let i-th ($i = 1, \ldots, K$) denote the silhouette image by a N-dimensional column vector, $\mathbf{X}_i$, of binary mask: 1/0 respectively indicates the foreground/background (**fg**/**bg**). N is the number of pixels in $\mathbf{B}$. Then, all the K samples form an *input space* Φ. Suppose a kernel function k is used to induce a nonlinear mapping φ from Φ to some *feature space* Ψ. The corresponding kernel matrix is $\mathbf{K} = [k(\mathbf{X}_i, \mathbf{X}_j)]_{K\times K}$ and it can be centralized via $\mathbf{K} = \mathbf{HKH}$, where $\mathbf{H} = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T$, $\mathbf{I}$ is an identity matrix, and $\mathbf{1}$ is a column vector of ones.

Kernel PCA space. Let $\lambda_1 \geq \ldots \geq \lambda_M > 0$ be the $M \leq K$ positive eigenvalues of $\mathbf{K}$ and $\mathbf{u}_1 \geq \ldots \geq \mathbf{u}_M > 0$ be the corresponding eigenvectors. The embedding

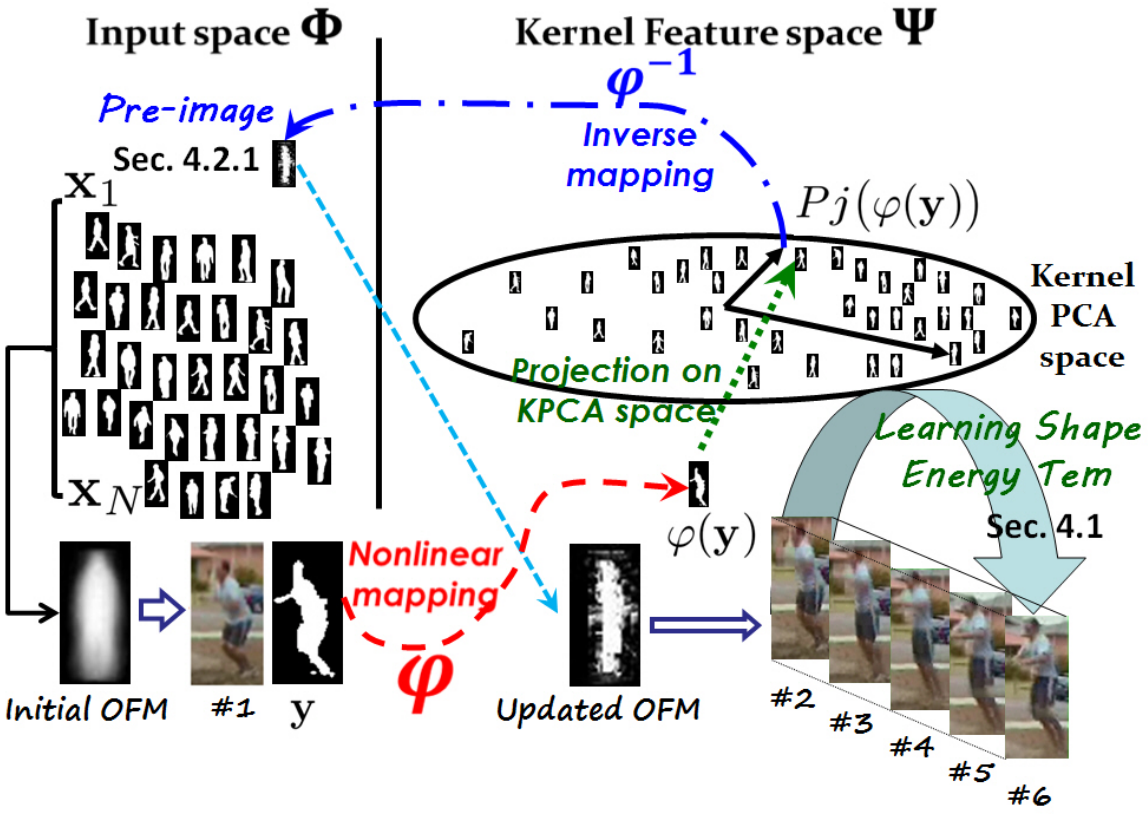


Figure 6.3: Learning a shape energy term and updating occurrence frequency map (OFM) via KPCA. A training set induces a KPCA space and an initial OFM: 1) a shape prior is learned in the KPCA space; 2) the initial OFM helps to generate the initial segment; 3) the segment's KPCA pre-image is set as the updated OFM for the next volume.

dimensionality M is set to the rank of $\mathbf{K}$, or a smaller value to ignore insignificant dimensions. Thus, the M principal components of the Kernel PCA space can be written as $1/\sqrt{\lambda_j}\sum_{i=1}^{K} u_{ij}\varphi(\mathbf{X}_i), j = 1, \ldots, M$. The details of the KPCA algorithm are omitted due to the space limit. Refer to [231] for details.

Shape energy. Now, let us define a testing silhouette's corresponding vector as $\mathbf{Y} = [y_1, ..., y_N]^T$. The idea of minimizing the shape energy within the Graph cuts

objective is minimizing the distance between the testing shape and the mean shape learned from the training data. However, the mean in Ψ can not be explicitly computed [231]. In practice, it is sensible to drive the mapped testing shape toward its projection on the KPCA space [229]. Thus, a shape energy $S^{\Psi}(\mathbf{Y})$ in $\Psi$is defined as the squared distance between $\mathbf{Y}$'s mapping $\varphi(\mathbf{Y})$ and the further projection $Pj\big(\varphi(\mathbf{Y})\big)$ on the KPCA space. From [229], it can be expressed as

$$\begin{aligned} &S^{\Psi}(\mathbf{Y}) = \|\varphi(\mathbf{Y}) - Pj\big(\varphi(\mathbf{Y})\big)\|^2 = \\ &k(\mathbf{Y},\mathbf{Y}) + \frac{1}{K^2}\mathbf{1}^T\mathbf{K}\mathbf{1} - \frac{2}{K}\mathbf{1}^T\mathbf{k}_Y + \tilde{\mathbf{k}}_Y^T\mathbf{M}\tilde{\mathbf{K}}\mathbf{M}\tilde{\mathbf{k}}_{\mathbf{Y}} - 2\tilde{\mathbf{k}}_{\mathbf{Y}}^{\mathbf{T}}\mathbf{M}\tilde{\mathbf{k}}_{\mathbf{Y}} \end{aligned} \tag{6.1}$$

and transform it back to the input space φ as

$$S(\mathbf{Y}) = -2\sigma^2 \log\Big(1 - \frac{1}{2}S^{\Psi}(\mathbf{Y})\Big) \tag{6.2}$$

where $(\mathbf{k}_Y)_{K\times 1} = [k(\mathbf{Y},\mathbf{X}_1),...,k(\mathbf{Y},\mathbf{X}_K)]^T$, $(\tilde{\mathbf{k}}_Y)_{K\times 1} = \mathbf{H}(\mathbf{k}_Y - \frac{1}{K}\mathbf{K}\mathbf{1})$, and $\mathbf{M}_{K\times K} = \sum_{i=1}^{M}\frac{1}{\lambda_i}\mathbf{u}_i\mathbf{u}_i^T$. where the subscript such as $K\times 1$ denotes the dimension of a matrix. $S^{\Psi}(\mathbf{Y})$ will be used to compute the KPCA's pre-image, and $S(\mathbf{Y})$ will be used in the segmentation objective.

6.1.3 Seed label initialization and propagation

This section aims to construct seed maps for every τ testing frames online. Let $\mathbf{Q} = [q_1,...,q_N]^T$ denote the pixel's color value in one testing frame's $\mathbf{B} = \{\rho_1,...,\rho_N\}$, and $\mathbf{Y} = [y_1,...,y_N]^T$ denote the wanted silhouette image of $\mathbf{B}$. To generate a seed map, it is preferred to threshold the posterior probability distribution $\mathbf{P}(\mathbf{Y}|\mathbf{Q})$ of the

fg/bg color distribution $\mathbf{P}(\mathbf{Q}|\mathbf{Y})$ and a statistical prior $\mathbf{P}(\mathbf{Y})$ to assign labels y. Note that $\mathbf{P}(\cdot)$ is used to denote a matrix corresponding to the image patch $\mathbf{B}$, and use $p(\cdot)$ to denote a matrix element corresponding to a pixel.

6.1.3.1 Occurrence Frequency Map (OFM)

Initialization. For training samples of silhouette images, the probability $p(y = 1)$ of a pixel being fg equals to the frequency of that pixel having an intensity value 1, namely the pixel value in the Occurrence Frequency Map $\mathbf{P}_{occ}$. See Fig. 6.4 (b) for an illustration. Therefore,

$$\mathbf{P}_1(\mathbf{Y}) = \mathbf{P}_1^{occ} = \textstyle\sum_{i=1}^{N} \omega_i \cdot \Gamma_i,$$

where Γ_i is the matrix of the i-th sample, which is obtained from the online retrieval; ω_i is the weight assigned to Γ_i and obtained via logistic regression in Sec. 6.1.1; the superscript *occ* and subscript 1 denote occurrence and frame #1.

Propagation. The KPCA projection $Pj\big(\varphi(\mathbf{Y})\big)$ produces a shape most consistent with the training set [229]. [222] takes the KPCA projection's pre-image $\hat{\mathbf{Y}}$ to be a prior occurrence map via approximate inverse mapping, which induces promising results. Besides, the temporal coherence ensures the smoothness of the shape changes between adjacent frames, and even in a *temporal sliding window* across τ frames. Thus, from frame #2, once the previous frame's silhouette $\mathbf{Y}_t$ is available, $Pj\big(\varphi(\mathbf{Y}_t)\big)$'s pre-image $\hat{\mathbf{Y}}_t$ can be set as a prior $\mathbf{P}_{t+1}^{occ}$ for the subsequent τ frames: $\mathbf{P}_{t+i}(\mathbf{Y}) = \mathbf{P}_{t+i}^{occ} = \hat{\mathbf{Y}}_t$, where $i = 1, ..., \tau$; $t = j \cdot \tau + 1$ and $j = 0, 1, 2, ...$ Notably, $\hat{\mathbf{Y}}$

is not easily computed, for φ is implicit and nonlinear. By referring to Sec. 4 in [222] and Sec. 3 in [231], an approximate pre-image is computed as $\hat{\mathbf{Y}} = \frac{\sum_{i=1}^{N} \alpha_i \mathbf{X}_i}{\sum_{i=1}^{N} \alpha_i}$, where $(\alpha_i)_{N\times 1} = \tilde{\gamma}_i\big(1 - \frac{1}{2}S^{\Psi}(\mathbf{Y})\big)$, $\quad (\tilde{\gamma}_i)_{N\times 1} = \gamma + \frac{1}{N}(1 - \mathbf{1}^T\gamma)$,

$\gamma_{N\times 1} = [\mathbf{u}_1, ..., \mathbf{u}_N][\frac{1}{\sqrt{\lambda_1}}\mathbf{u}_1, ..., \frac{1}{\sqrt{\lambda_N}}\mathbf{u}_N]^T\tilde{\mathbf{k}}_{\mathbf{Y}}$, $S^{\Psi}(\cdot)$ is defined by Eqn. 6.1,

and $\mathbf{1}, \mathbf{u}_i, \lambda_i, \tilde{\mathbf{k}}_{\mathbf{Y}}$ are defined in Sec. 6.1.2. Essentially, the pre-image $\hat{\mathbf{Y}}$ is a linear combination of the training data $\mathbf{X}_i$, in a similar fashion with K-means or GMM, while the coefficients are computed differently. Please see [231] for the intuition and details. Now, for all frames, $\mathbf{P}(\mathbf{Y})$ has been obtained.

6.1.3.2 Posterior Probability Map (PPM)

Next, a coarse segment $\mathbf{C}$ is acquired by thresholding $\mathbf{P}^{fg}$. Using GMM on the pixels marked as fg/bg seeds, each fg pixel's likelihood $p(q|y = 1)$ is modeled within $\mathbf{C}$, while each bg pixel's likelihood $p(q|y = 0)$ is modeled outside $\mathbf{C}$ but inside $\mathbf{B}$. The initial $\mathbf{C}$ is achieved via level set. With the *priors* $p(y = 1)$, $p(y = 0)$ and the likelihoods $p(q|y = 1)$, $p(q|y = 0)$ available, the *posterior probabilities* $p(y = 1|q)$ and $p(y = 0|q)$ can be computed:

$$p(y|q) \propto p(q|y) \cdot p(y) \tag{6.3}$$

With all $p(y|q)$ computed and normalized, the *Posterior Probability Map* $\mathbf{P}(\mathbf{Y}|\mathbf{Q})$ is formed, on which thresholding is conducted to set up a *trimap* consisting of fg, bg and the unknown.

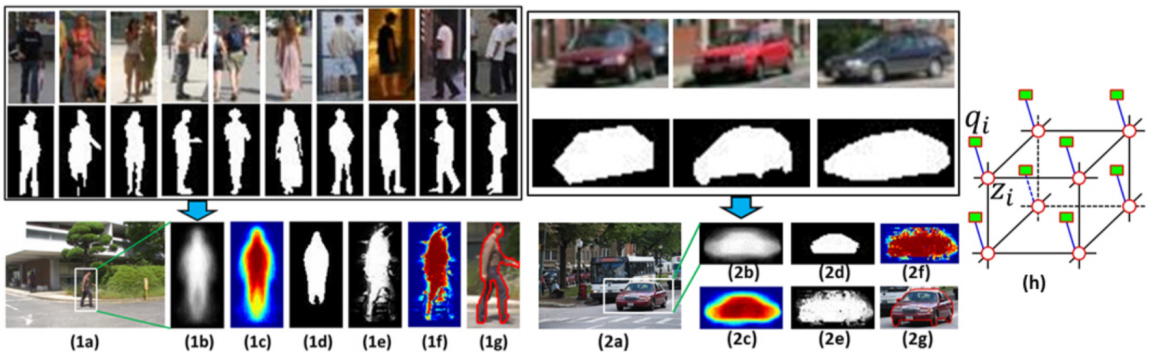


Figure 6.4: Seed initialization for a pedestrian sequence (1) and *Hopkins155*-part6-car10 (2). The top row displays retrieved training samples. Second row: (a) Selected object in the initial frame as a query, (b) Learned OFM corresponding to the query, (c) OFM shown as a heat map, (d) Coarse segment by thresholding OFM, (e) Learned PPM denoting the probability of each pixel being fg, (f) PPM shown as a heat map, (g) Graph cuts result, (h) The graphical model illustrating the 3D graph, in which z_i is a binary variable denoting pixel i's label, and q_i is the color value of pixel i.

6.1.3.3 Spatial segmentation via 3D Graph Cuts

In this section, the object is segmented volume by volume, or called tublet by tublet, which is a 3D graph formed by a τ-frame short sequence of tracked $\mathbf{B} = \{\rho_1, ..., \rho_N\}$. Each voxel ρ_i contains 26 neighbors. All the unordered neighboring voxel pairs are stored in (ρ_i, ρ_j) in a set Π. Let $\mathbf{y} = [y_1, ..., y_N]^T$ denote the vector of the silhouette in one frame, then the silhouette results can be denoted as a $N \times \tau$ matrix $\mathbf{Z} = [\mathbf{y}_1 \ ... \ \mathbf{y}_\tau]$. The graph energy function is constructed as

$$E(\mathbf{Z}) = \mu_1 R(\mathbf{Z}) + \mu_2 T(\mathbf{Z}) + \mu_3 S(\mathbf{Z}) \tag{6.4}$$

where $R(\cdot)$ and $T(\cdot)$ are the *region term* and *boundary term*, respectively. Our contribution is the holistic *shape term* $S(\mathbf{Z}) = \sum_{i=1}^{\tau} S(\mathbf{y}_i)$, where $S(\mathbf{y}_i)$ has been defined in Eqn. 6.2, and μ_1, μ_2, μ_3 specify the weights. $R(\cdot)$ and $T(\cdot)$ are defined as

$R(\mathbf{Z}) = \sum_{i=1}^{N \cdot \tau} R_\rho(z_i)$ and $T(\mathbf{Z}) = \sum_{\{\rho_i,\rho_j\} \in \Pi} T_{\{\rho_i,\rho_j\}} \cdot \delta(z_i, z_j)$, where

$$R_{\rho_i}(z_i) = -\ln p(z_i|q_i) = -\ln p(q_i|z_i) - \ln p(z_i) \quad (6.5)$$

$$T_{\{\rho_i,\rho_j\}} = \frac{k(q_i, q_j)}{\|\rho_i - \rho_j\|} \quad (6.6)$$

$$\delta(z_i, z_j) = \begin{cases} 1 & z_i \neq z_j \\ 0 & z_i = z_j \end{cases}$$

are the voxel-level *region unfitting penalty*, *boundary discontinuity penalty*, and *neighborhood binary regulator*, respectively. Eqn. 6.5 is a negative log-likelihood, penalizing an assignment of fg and bg voxels during the optimization according to how well they fit the images' color distributions. It is derived using Eqn. 6.3, where z replaces y and $\mathbf{P}(\mathbf{Z}|\mathbf{Q})$ denotes the histogram for fg color distribution when $z = 1$, and similarly for bg when $z = 0$. Unlike [217]'s hard assignment of seeds, a statistical occurrence frequency is incorporated as a prior and set the posterior as a probabilistic color distribution. In Eqn. 6.6, $k(\cdot, \cdot)$ is a kernel. Eqn. 6.6 penalizes a cut across voxels, according to their similarity relative to the distance. Then the model minimizes Eqn. 6.4 using the *Max-flow/min-cut* algorithm [233]. Except for the initial frame, 3D Graph cuts are performed *once* for all the frames in a volume.

6.1.4 Experiments

The cross validations is performed for parameter selection and use Gaussian kernels in KPCA and Graph cuts. For GMM, the model adopts the EM algorithm

initialized by K-means. In default, the Gaussian numbers of fg and bg are set to 4 and 5 respectively; τ is set to 5; μ_1, μ_2, μ_3 are 0.4, 0.4, 0.2 respectively; the margin accounts for 20 percent of the width/height of the original bounding box. The whole system runs at about 3$\sim$6 fps.

Analysis of top-down constraints. In our method, there are three top-down constraints: the rough location, the shape prior, and seed labels. As shown in Table 2, there are five variations in total. The version without tracking but with seed updating does not exist, for seeds are predicted by modeling the fg and bg within a bounding box. First, an experiment is performed on Youtube-seq48 and calculate the precision-recall, as shown in Fig. 6.5 and Table 3. A higher precision implies less distraction, namely fewer outliers and fewer false positives; a higher recall implies a more complete segment, namely fewer false negatives. In Fig. 6.5, a comparison between (b) and (a) indicates that tracking induces less distraction. Namely, in Table 3, (b) has a remarkably higher precision than (a). Comparisons of (e) *v.s.* (d) and (c) *v.s.* (b) indicate that seed label update also induces less distraction. See (e) *v.s.* (d) and (c) *v.s.* (b) in precision. Comparisons of (e) *v.s.* (c) and (d) *v.s.* (b) indicate that a shape prior induces a more complete segment, namely (e) has a remarkably higher recall than (c) and similarly for (d) *v.s.* (b). Table 4 presents the F-measure score, which is defined as the *harmonic mean* of precision and recall. The larger F-measure value, the better performance.

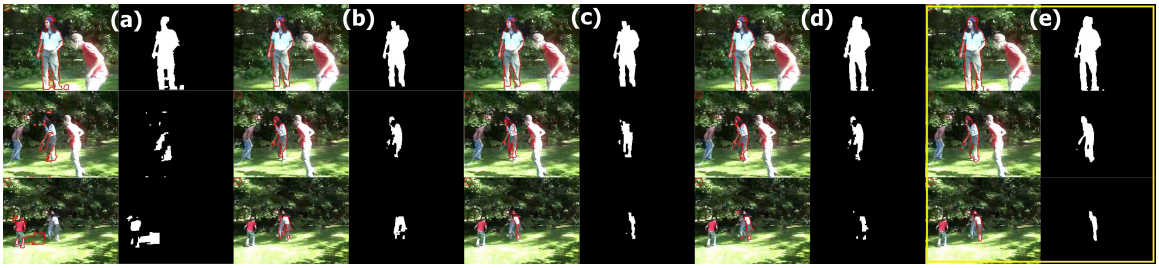


Figure 6.5: Comparison of 5 versions of our method on seq48 [1]. From left to right: the segmentation results and corresponding silhouette images generated by versions (a), (b), (c), (d), (e) respectively. From top to bottom: Frame 1, 31, 61, respectively.

	No update	Seed update
No track, no shape term	(a)	×
Tracking, no shape term	(b)	(c)
Tracking, shape term	(d)	**(e)**

Table 6.2: Five versions of our method.

Method	Ave. Prec.	Ave. Rec.	F-score
(a)	0.13	0.25	0.17
(b)	0.49	0.47	0.48
(c)	0.62	0.53	0.57
(d)	0.51	0.64	0.57
(e)	**0.72**	**0.68**	**0.70**

Table 6.3: Test on Youtube-seq48 [1].

Overall evaluation and comparison. Our method is evaluated on the whole Youtube dataset [1] and compare it with two related methods (see Table 4). Our full method achieves competitive performance with the state-of-the-art [221], which learns top-down shape priors from a set of pre-selected templates and uses them to guide a bottom-up level-set based segmentation. [221] can only segment frames with pedestrian detections, and its shape prior is limited to their fixed and small training set. In our method, the dynamic training set's size N is set to 500 for this "*person*" task, which is the most interested and more challenging than most other object classes

Method	Ave. Prec.	Ave. Rec.	F-score
Our method's partial version (a)	0.15	0.27	0.19
Our method's partial version (b)	0.35	0.38	0.36
Our method's partial version (c)	0.65	0.41	0.50
Our method's partial version (d)	0.41	0.63	0.50
Our method's full version (e)	0.71	**0.80**	**0.75**
Niebles *et. al.* 2008	0.57	0.44	0.50
Niebles *et. al.* 2010	0.74	0.75	0.74

Table 6.4: Evaluation on the Youtube dataset [1].

(*e.g.*, *car* in Fig. 6.4) that [221] cannot handle. Actually, a series of number are tested, *i.e.*, $N = 50, 100, 150, ..., 1000$. Generally, a larger N induces a higher F-score, while the increase of F-score is slight after $N = 500$. See our supplemental video for more results. Note that segmentation is performed on the enlarged bounding boxes from tracking.

6.1.5 Summary

A flexible and generic framework is presented to segment moving objects in videos using priors learned online from a large-scale Web image set. Its performance is evaluated on the Youtube dataset [1], and shown to be competitive with the state-of-the-art [221] that relies on offline modeling. Our insight is to learn knowledge online

from the rich Web data. The future work includes one-shot multi-object segmentation [224, 227], and simultaneous tracking and segmentation, which is expected to help preserve temporal coherence and in turn benefits seed label propagation. Further, our results may enrich *LabelMe*, which is growing with more and more annotated images and video snapshots. As suggested in [213], performance can be improved by domain adaptation between images and videos. It is worth trying to directly retrieve videos as training data, from sources such as *LabelMe video* with video-level annotations.

Other top-down priors can also be employed, such as attention. It is also worth discussing on segmenting multiple instances simultaneously. The use of shape, color and entropy features induces obtaining representative training samples. Although sample retrieval and weight learning cost some extra time, the segmentation method itself is computationally efficient, for it only processes the bounding box patch and carries out one-short segmentation for all frames in a volume.

The choice of specific model for each step is flexible. Actually, this chapter focuses on how models are naturally connected. In the context of establishing such a highly-automated app, a combination of specific models is necessary. The use of bonding box makes it feasible to fuse tracking, prior/seed learning and segmentation. Besides, it brings benefits of less distraction, lower computation cost, and robustness to the global motion.

6.2 Action quality assessment using segmental 3D ConvNet

Real-time sports video broadcast has reached mature commercialization with solutions ranging from image processing such as video coding, compression and communication to computer vision such as multiple camera synchronization used by EyeVision 360 for the early 2000s Super Bowl. Gradually, the industry realizes that video replay can be used for displaying the game process as well as affecting the progress in a hopefully positive way such as the goal-line or off-line detection by checking if the ball or player is on the homography plane. Now, Video Assistant Referee (VAR) has been adopted by FIFA for the 2014 & 2018 World Cup and populated into NFL and European premier leagues. While its value in real games is debatable, internal training has benefited from the real-time statistics provided by a couple of companies such as ©Sportradar, STATS, Gracenote, Sportlogiq and Signality.

Compared with combat sports, evaluation accuracy is relatively easier to be achieved in performing sports such as figure skating, diving and gymnastics. Obtaining accurate performance score is important to effectively maintain the fairness and avoid corner cases where human referees may either make mistakes or have bias. Traditionally, referees also watch the slow-motion replay to manually consolidate a score from the subjective evaluations on each criterion according to a metric.

6.2.1 Action quality assessment related works

The word quality and skill are used interchangeably in this chapter as they refer to the same concept when describing an action. They have strong connections to photography, image and video quality which measures the human perceived degradation through the photograph, image and video compared to seeing the captured scene ourselves. Similarly, referee perceived degradation might happen during the performance of a certain action compared with the expectation of a perfect action. For example, there is a great desire in healthcare to assess surgical actions [234] which have strong temporal structures.

In terms of sports actions, there exist two pioneering works at ACCV 2010 [235] and ECCV 2014 [236]. Wnuk and Soatto run SVR on the bag of SIFT, HoG and HoF features. For the first time, they collect a diving scoring dataset FINA09 named after Int'l Swimming Federation. It contains 12 different divers each of who performs 6 unique dives from the mens 3-meter springboard diving event at the 2009 FINA World Championships. However, FINA09 is not available now.

At ECCV 2014, authors of [236] run SVR on human poses to score their MIT Diving dataset upon which authors of [62] build the UNLV-Dive dataset. The C3D+SVR approach of [62] has shown significant improvements over previous works [237] using the approximate entropy features. In detail, the authors of [236] used human pose features for selected salient frames and used support vector regression (SVR) to score the videos. In the work of [237], better representations are used by calculating the

approximate entropy features for each frame. Furthermore, Parmar *et. al.* [62] proposed several approaches based on the C3D features [61]. SVR, LSTM and LSTM + SVR are used to score actions. C3D+SVR has shown significant improvements over previous state-of-the-art by that time.

Other sports action such as figure skating is also tested in [62, 236]. Actions of the gymnastic vault are also included in [62]. While more tests on different sports help demonstrate the generalizability of the proposed model, this section turns to focus on the generic design of the model and identify any specific domain knowledge that might play the key role during the learning process. Our proposal and analysis are hoped to open a door to wide applications in various domains such as exercise [238] and surgical actions [234]. And also, please see references in [62, 234, 236, 237, 239] for the broad interests in surgical and how-to videos. All these action quality assessment works are motivated by solving the same problem that the manual assessment has to be done by experts so even crowd-sourcing is not feasible.

Deep learning seems a good fit to automate this end-to-end process of estimating a score for a given video [240]. Indeed, Parmar and Morris [62] score diving by applying C3D-LSTM [61] on full videos in the dataset UNLV-Dive they extend from MIT-Diving [236] collected from the 2012 London Olympics 10-meters platform diving. As shown in Fig. 6.6, the performance is boosted with the more powerful P3D [241] using less memory (see Sec. 6.2.3). Unfortunately, it is hard to understand which element plays the key role in this line of works.

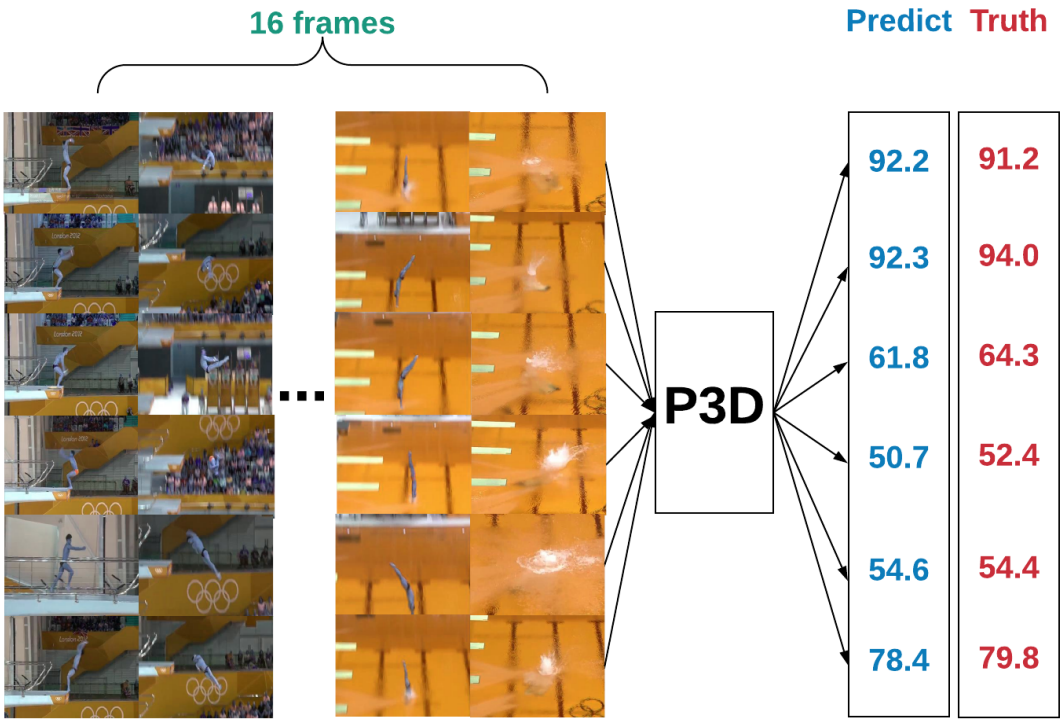


Figure 6.6: Adding a fully-connected layer on top of the 2nd last layers of P3D for regression. A training set of 16-frame clips sampled from raw videos of the UNLV-Diving dataset are input into the revised P3D network equipped with weights pre-trained on the Kinetics dataset. The scores predicted by the network (in blue) are compared with the ground truth in red.

In our study, it is found that full-video P3D (see Sec. 6.2.3) performs similarly with looking at the water spray only while P3D on the video except the water spray works poorly (see Sec. 6.2.6.3). Although smooth entering into water is the key to high scores, there must be a way for the network to gain knowledge from previous stages. A sensible way is to apply P3D on each stage and then fuse the features or subscores instead of feeding the entire video into P3D. This model is carefully implemented and achieves the state-of-the-art performance even with the full-video label used as the inaccurate stage-wise label (see Sec. 6.2.4). It is shown that temporal

action segmentation can be done with few efforts in Sec. 6.2.5. While our algorithm is not the first to score sports actions, to our knowledge it is the first to score them stage by stage.

Action quality assessment in our case is to predict a score s given a video $\mathbf{V}$ of one diving performance. As a supervised learning model, CNN is supposed to learn a mapping $f(\cdot)$ from $\mathbf{V}$ to s from training data such that $s = f(\mathbf{V})$. While it looks like that the action quality score is a function of the action video, essentially the video is a representation of the player's skill. As a result, there also exists $\mathbf{V} = g(s)$ where $g(\cdot)$ is a generative function: given a certain skill s, the generated action is recorded in the video $\mathbf{V}$. However, this section tries to learn the underlying representation of the skill s from the video $\mathbf{V}$. Namely, if the mapping $f(\cdot)$ learned by CNN is good enough, it well characterizes the inverse video generation process as $s = f(\mathbf{V}) = f(g(s))$.

This section extends our idea of regularizing the state-of-the-art classification model to regression in [240] yet for human actions. Moreover, the domain knowledge is taken such as action segments predicted by TCN to induce the sensible approach of scoring each stage and then fusing the subscores or features in the end.

6.2.2 Intuition of 3D Convolution Factorization

The effectiveness of factorizing 3D convolution as a product of a spatial 2D convolution and a temporal 1D convolution is shown empirically in the works of Pseudo-3D CNNs (P3D) [241], Inflated 3D CNNs (I3D) [63] and Factorized Spatio-Temporal

CNN ($F_{ST}CN$) [242]. However, there does exist very limited progress of theoretical analysis of the separability. This section briefly touches upon the underlying principle and intuitive correctness of dividing a separable 3D convolution filter as a combination of a 2D convolution filter and a 1D convolution filter. For a three-layer perceptron with one hidden layer, the network's optimization is essentially solving

$$\min_{\mathbf{U},\mathbf{V}} \|\mathbf{Y} - \mathbf{U}\mathbf{V}^T\mathbf{X}\|_F^2 + \theta(\mathbf{U}, \mathbf{V}) \tag{6.7}$$

where the weights from the input layer to the hidden layer are arranged in the vector $\mathbf{V}$ and the weights from the hidden layer to the output layer are arranged in the vector $\mathbf{U}$. F denotes the matrix Frobenius norm. As a result, the hidden layer $\mathbf{z} = \mathbf{V}^T\mathbf{x}$ and the out layer $\mathbf{y} = \mathbf{U}\mathbf{V}^T\mathbf{x}$. In the recent literature, there are theoretical guarantees of global optimality with a specific regularization $\theta(\cdot)$ [243] and also non-linear functions as long as they are positive homogeneous such as ReLU [244]. Intuitively, if it is provable for the optimality of three-layer 2D perceptron with a single hidden layer and a certain regularizer, then it is possible to adapt the proof to deep multi-layer 2D perceptron with multiple hidden layers.

For videos, there is a time axis. There is $\mathbf{I}(\mathbf{x}, t) = \sum \mathbf{u}_i(\mathbf{x}, t)\mathbf{v}_i = \sum_{i,j} \mathbf{u}_i(\mathbf{x})\mathbf{v}_j(t)c_{i,j}$ where $\mathbf{u}_i, \mathbf{v}_i$ are the per-node weights of $\mathbf{U}$ and $\mathbf{V}$, respectively. The separability without convolutions is that the adaption of the Eqn. 6.2.2 to 3D can be represented as the product of the spatial basis, the temporal basis and the coefficients:

$\theta = \sum_{i,j} \|\mathbf{u}_i\|\|\mathbf{v}_j\||w_{i,j}|$. Given those connections, intuitively if it is provable for the optimality of deep multi-layer 2D perceptron, then it is possible to adapt the proof to deep multi-layer 3D perceptron or simply called deep 3D networks.

Furthermore, for a convolution network, there is $\mathbf{I}(\mathbf{x}) = \sum_{i=1}^{r} \mathbf{u}_i(\mathbf{x}) * \mathbf{v}_i(\mathbf{x})$ where $*$ denotes the operation of convolution. The operation of u_i convolving with v_i can be written as the matrix convolution in the dimension of $n_{pixels} \times (n_{pixels} * size_{filter})$ to induce a 4D tensor:

$$f_i(x, y, t) = \mathbf{u}_i(\mathbf{x}, \mathbf{y}, t) * \mathbf{I}(\mathbf{x}, \mathbf{y}, t) = \mathbf{u}_i(\mathbf{x}, \mathbf{y}) \cdot \mathbf{v}_j(t) * \mathbf{I}(\mathbf{x}, \mathbf{y}, t) \tag{6.8}$$

Given the connection from matrix product to matrix convolution, if it is provable for the optimality of deep 3D networks, it is feasible to adapt the proof to deep 3D CNNs.

6.2.3 Action quality assessment using full-video P3D

The Pseudo 3D (P3D) network has been shown in [241] to be effective on action classification since both appearance and local motion are modeled. The last layer of P3D is replaced by a fully connected (FC) layer with dropout so that the model originally trained for classification can be used for regression.

In C3D [61], the 3D convolution layers have kernels with size $3 \times 3 \times 3$, which makes it extremely hard to train C3D. In P3D, the $3 \times 3 \times 3$ separable convolution

kernels are represented as a combination of $3 \times 3 \times 1$ and $1 \times 1 \times 3$ kernels. This allows the networks to perform the 2D and 1D convolutions separately. Similar to ResNet, P3D has these convolution layers ensembled into the residual units. There are three types of units depending on the strategy of fusing 2D and 1D layers:

P3D-serial:

$$x_{t+1} = x_t + T(S(x_t)) \tag{6.9}$$

P3D-parallel:

$$x_{t+1} = x_t + T(x_t) + S(x_t) \tag{6.10}$$

P3D-composition:

$$x_{t+1} = x_t + S(x_t) + T(S(x_t)) \tag{6.11}$$

where the x_t and x_{t+1} represent the input and output of the t-th residual unit, and $S(\cdot)$ and $T(\cdot)$ represent the 2D (spatial) and 1D (temporal) layers respectively. P3D also applies the bottleneck design in each Residual unit: $1 \times 1 \times 1$ convolution layers are placed at the beginning of a unit to reduce the input dimensions and at the end to increase output dimensions.

6.2.4 Action assessment using Segment-level P3D

As shown in Fig. 6.7, P3D can be trained on 16 frames centering around the middle frame of each stage independently. Using the trained models, the predicted sub-score can be obtained and feature for every stage. If features are used, four

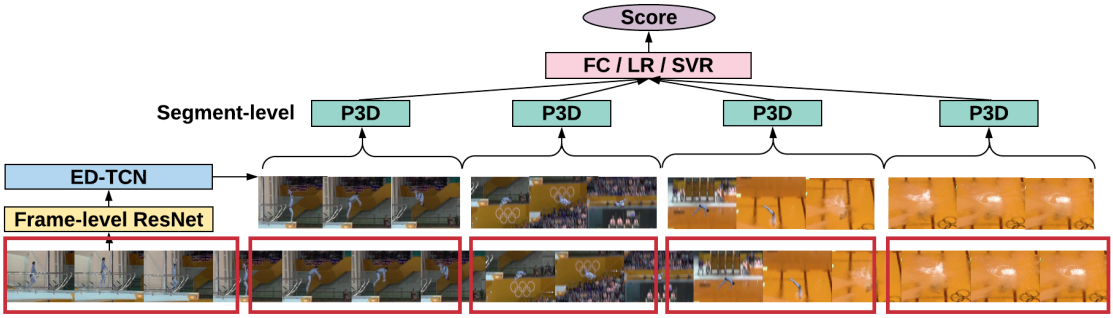


Figure 6.7: S3D is a stacked regressor with multiple pre-trained P3D networks. To perform the S3D, firstly ED-TCN is used to generate segments from full videos. Then four P3Ds (one for each segment) are trained to get feature representations. Lastly, the fully-connected regression layer, SVR or LR is used to predict the final score based on the average of all stage-wise features.

features are averaged to get a single feature. If sub-scores are used, four sub-scores are arranged as 4-dimensional feature vector (one score for each stage). Over the train-split videos, there is a training set of (x, y) to learn a mapping from the feature x to the ground-truth score y using LR and SVR so that y can be predicted given an unseen x in the test-split using the mapping. Indeed, diving actions can be segmented into the following 3 stages:

(i) Jumping: leaving platforms to hands closing to the body.

(ii) Dropping: from hands firstly holding close to body as a V shape until releasing the hands to revert body.

(iii) Entering into the water: from releasing hands to revert body until body fully entering into the water.

In addition, per video there are a beginning stage capturing the players making prepa-

rations and an ending stage capturing the pool with water spray decaying. The beginning is not scored but the ending is. Thus, there are four stages to score. Now, the induced algorithm is summarized in Algorithm 1.

Algorithm 7: Using S3D to score diving performance.

(1) Use TCN to segment the video into five stages (Stages 0-4).

(2) On each stage from Stages 1-4, fine-tuning a P3D CNN separately.

(3) Use four models to extract features for every video (one model on one stage, so there are four 1D features for one video)

(4) Average these four feature vectors into a single feature vector.

(5) Use SVR ot LR to learn a mapping from the features to the score of videos.

Notably, fine-tuning the segment-level P3D with the full-action label is weakly-supervised learning. There is a weak relation between the segment-level label and the full-action label: when the full action is good, each segment cannot be too bad; when a segment is certainly bad, the full action could hardly be good. However, when the full action is bad, it is not sure about which segment is bad. When a segment is good, the full action cannot be guaranteed to be good. Sec. 6.2.6.2 empirically demonstrates that it is doable to live with such a sub-optimal learning schema. It turns out that it lives very well. Namely, the fusion of multiple weakly-supervised segment-level P3Ds performs way better than the fully-supervised full-video single P3D. The reason is likely to be that fusing multiple P3D outputs using regression is essentially ensemble learning or more specifically bagging. The idea of bootstrap

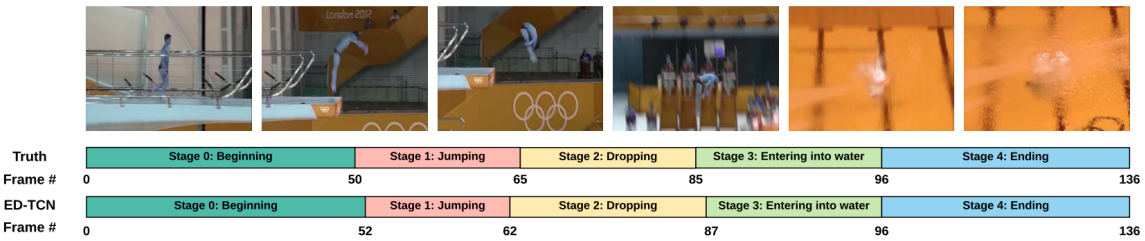


Figure 6.8: Visualization of TCN segmentation result on video # 186. The beginning stage is not passed into P3D as it is standard.

aggregating is to train several models separately and then have all the models vote on the output for test examples. Indeed, the regression serves as the model voting or averaging.

6.2.5 Temporal localization/segmentation via TCN

The task of temporal segmentation in our case study is to classify frames into 5 classes with the intra-class continuity constraint. Taking the frame-level 2D CNN features as inputs, TCN can return five segments (one preparation stage, three action stages and one background stage) for a diving video. Suppose an input video has K frames and the output feature is D-dimensional, then the input to TCN can be denoted as $\mathbf{X}_0 \in \mathbb{R}^{D \times T}$ where the subscript is the count of layers traversed till now (below $l = 0, 1, ...$ are used for layer). For the ED-TCN [42], the temporal convolution is represented as

$$\mathbf{X}_l = f(\mathbf{W}_l * \mathbf{X}_{l-1} + \mathbf{b}) \tag{6.12}$$

where $\mathbf{X}_l \in \mathbb{R}^{N_l \times T_l}$, $N_0 = D$, $T_0 = K$. The convolution filters are parameterized by

$\mathbf{W} = \{\mathbf{w}_i\}_{i=1}^{N_l}$, $\mathbf{w}_i \in \mathbb{R}^{d_l \times N_{l-1}}$ and $b \in \mathbb{R}^{N_l}$ for N_l being the number of convolution filters at l-th layer, T_l being the number of features, d_l being the filter length at l-th layer and $f(\cdot)$ being the activation function.

6.2.6 Experiments

In this section, experimental results are presented. The project repository [1] has been set up. Our segment labels are available there.

6.2.6.1 Implementation details

P3D. The implementation uses a PyTorch implementation of P3D-199 model[2] with weights pre-trained on Kinetics and revised it into a regression model. The P3D-consecutive model is trained and tested on 16 consecutive frames that randomly selected from the entire video. For the P3D-sampled model, frames are sampled from the video with equal space. The selected 16 frames are then resized into 160×160 to be input into models during training and testing. Residual units are in the order of P3D-serial P3D-parallel P3D-composition. Dropout with rate 0.5 is applied to the top FC layer. The MSE loss is used as the loss function in training. Adam is used with the learning rate of 0.0001 as our optimizer. Models are trained for 90 epochs with learning decay factor of 0.1 for every 30 epochs.

Video sampling. As P3D is designed to process clips of 16 frames, fine-tuning the

[1] `https://github.com/eglxiang/diving-score-1`
[2] `https://github.com/qijiezhao/pseudo-3d-pytorch`

pre-trained P3D model needs clips of 16 frames. There are three effective strategies for sampling 16 frames from a video. Sec. 6.2.6.3 compares them. Firstly, normally a 3D network needs to be trained for many epochs on small datasets. A good strategy turns out to be randomly stopping a sliding window of 16 consecutive frames along the temporal axis, which not only keeps the action smooth and coherent but also introduces certain randomness during each epoch. Thus, it also augments the data. Models trained and tested using this strategy are named as *P3D-consecutive*. There is information loss during the random consecutive sampling. P3D can only read a 16-frame clip per video and thus can hardly see all the four stages relevant with scoring, while all influence the score. A bad case is that the first stage is sampled. Secondly, as video summarization, equally spaced sampling collectively represents the video and cover all stages, though sacrificing the temporal coherence. This model is called *P3D-spaced*. Lastly, if the center frame of each stage is given, the P3D input is chosen to be the 16 frames that are centered at the middle of each stage and have a spacing of 1. They serve as a summarization of this stage and will be input into the *P3D-center* model of that stage.

ED-TCN. Since the video length varies, Zeros are padded at the end to make them have identical 160 frames. ResNet is used to extract features per frame so the input dimension is 2048×160. The model is trained for 30 epochs with RMSProp.

6.2.6.2 Dataset and evaluation metric

Parmar *et al*. extended the MIT diving dataset [236] into a larger diving dataset UNLV-Dive with 370 videos [62]. Each is about 4 seconds long from the (semi-)final of the 2012 Olympics platform diving games. An overall score in the range of [0, 100] and a difficulty level are given for each video.

Consistent with previous works [62, 236], the Spearman rank correlation $\rho = cov(R_p, R_g)/\sigma_{R_p}\sigma_{R_g}$ is used as our evaluation metric where R_p and R_g denote the ranking of the per-video predicted score p and the ground-truth score g, $cov(R_p, R_g)$ is the covariance and σ_{R_p}, σ_{R_g} are the standard deviations. The correlation is 1 if two variables have an identical rank and -1 if having a fully opposed rank. Between -1 and 1, the higher the correlation, the similar the ranks of the two variables. A higher Spearman correlation is expected for a better prediction.

Notably, segmentation labels have been annotated for this dataset. Each video is segmented into five stages (beginning, jumping, dropping, entering into water, ending) by labeling the four indexes of the frame where the stage state transits. Performances are measured by accuracy (the hit percentage).

6.2.6.3 Experimental results and analysis

Temporal segmentation. The ED-TCN is tested with a different number of layers (2 to 3) in both encoder and decoder. Based on the number of layers, a different number of filters are tested on each layer (from 256 to 1024) and filter length (from 3 to

# Layers	# Filters	# Filter Length	Accuracy (%)
2	[256, 256]	[5, 5]	95.9
2	[256, 512]	[5, 5]	96.3
2	[256, 512]	[3, 3]	95.2
2	[256, 512]	[10, 10]	**96.6**
2	[256, 512]	[3, 5]	96.1
3	[256, 512, 1024]	[3, 5, 10]	95.3
3	[256, 512, 1024]	[10, 10, 10]	96.1

Table 6.5: Performance of ED-TCN with a different parameters. The layers represent the number of layers in encoder or decoder (the total number should be that number times 2) The filters and filters length represent the number for each encoder layer.

Temporal model	Acc (%)
Bi-LSTM [245]	95.7
ED-TCN	**96.6**
Tricornet (TCN+Bi-LSTM) [246]	96.0

Table 6.6: Accuracy comparison of temporal classification.

10). The optimizer of TCN is RMSProp with default parameter settings. The models are trained with a batch size of 64 for 30 epochs. The performances of ED-TCN with different parameters are shown in Table 6.5. Accurate temporal segmentation is given by ED-TCN as shown in Table 6.6 compared with relevant competing models and

Figure 6.8 which compares the predicted segments with the ground-truth segments. However, stage-wise P3D on jumping works poorly, on dropping works just in the level of previous works, and on jumping and dropping combined works poorly again. One reason is that since jumping is normally standard, using the full-video label to train P3D on it can even confuse other stages. Conversely, stage-wise P3D on the stage of entering into water works better than the full-video P3D and once again works poorly when combined with previous stages. Unsurprisingly, the performance on the ending stage without seeing the player is comparable.

Action quality assessment. Table 6.7 compares the performance of various versions of P3D and previous works. Going top-down, ConvISA [247] is an early effort of unsupervised spatiotemporal feature learning way before UNLV-Dive [62]. Together with Pose+DCT [236] and ApEnFT [237], performances on UNLV-Dive are reported in [62] by the creator of UNLV-Dive which is built upon MIT-Diving [236] created by authors of Pose+DCT [236]. It is wise to cite [62]'s performance of spatiotemporal C3D and notice that adding LSTM even hurts the performance. Reasons include that LSTM can hardly learn temporal evolution through *clip*-level features.

Going down, the performance is poor for the full-video P3D using consecutive sampling, the straightforward P3D extension to regression, unlike its good performance on action recognition where P3D and C3D use consecutive sampling. Reasons are mainly about the way of sampling. Normally action category can be told even when a part of the video is seen. However, the skill score is influenced by the entire

process of diving action. As a result, seeing a part is unlikely to work except seeing the water spray which is exactly one criterion that human judge will look at and highly correlates with action skills. However, the full-video P3D using simple equally-spaced sampling simply beats all previous C3D or pose based works and justifies the power of P3D as long as the input representation is comprehensive. Note that P3D-spaced is more efficient than C3D as it does not compute features clip by clip.

Going further down, the state-of-the-art performance can be achieved by the proposed S3D with the fusion of P3D features or scores though trained using the inaccurate full-video labels. The variants come from choosing SVR or LR and choosing averaging per-segment features or a collection of subscores.

Another aspect is that the proposed approach performs stably as shown by the standard deviation. For the last four, there is no randomness as it is using a fixed number of frames around the center frames. For others, the randomness mainly comes from the video sampling. But they are still stable. The credit should be given to the early stopping: the training stops once the test correlation reaches an optimal value.

However, it is understandable that achieving a high Pearson correlation does not necessarily mean our model is ready to be deployed in real games. At the moment, it only means that our model is imitating closely to human referees on a specific dataset. The application scenario of the induced system is to assist the human referees such as correcting their scores when they are off too much. However, in our experiments, the scores given by human referees are used as ground-truth labels for training and

Methods	Correlation
Hierarchical ConvISA [247] (ICCV 2011)	0.19
Pose+DCT+SVR (best in [236], ECCV 2014)	0.53
Entropy feature ApEnFT [237] (BMVC 2015)	0.45
C3D+LSTM [62] (CVPRW 2017)	0.36
C3D+LSTM+SVR [62] (CVPRW 2017)	0.66
C3D+SVR (the best in [62], CVPRW 2017)	0.74
Our P3D-consecutive + FC regression, full video	0.43 ± 0.09
Our P3D-spaced + FC regression on full video	*0.80* ± 0.01
Our P3D-center + FC regression, jumping stage	0.49 ± 0.04
Our P3D-center + FC regression, dropping stage	0.60 ± 0.03
Our P3D-center-FC, jumping-dropping combined	0.47 ± 0.04
Our P3D-center + FC on entering into water stage	*0.82* ± 0.01
Our P3D-center + FC, videos except ending stage	0.56 ± 0.04
Our P3D-center + FC regression, ending stage	*0.77* ± 0.02
LR on scores output by stage-wise P3D-center-FC	*0.82* ± 0
SVR on score output by stage-wise P3D-center-FC	*0.84* ± 0
LR on average of stage-wise P3D-center features	*0.81* ± 0
SVR on average of stage-wise P3D-center features	**0.86** ± 0

Table 6.7: Pearson correlation comparison on official split-4.

testing evaluation. Namely, it is assumed that the scores given by human referees are all correct while part of our work's motivation is that human referees do sometimes misjudge, make mistakes or have biases. As it is hoped that our model is able to generalize to other videos, one of the important future works is to bring in label error tolerant mechanism to the model. Although this point sounds philosophical, not improving it indeed restricts how well such a model as ours can perform in real-world systems. However, 100% correct labels are hardly available given the subjective nature of this problem. It turns out to be interesting to observe when the model disagrees with the labels, which are the cases making the Pearson correlation below 1. In this sense, our model's 0.86 correlation gives a good standing of our model compared with other models tested on this dataset. It does not imply too much for real-world applications. When the model has the access to the 100% correct labels or bring in a certain error-tolerant learning mechanism, the performance could be even better. First, the trained model's prediction power is restricted by the label quality of training data. Second, when the human referee label is incorrect, those test cases where the model might predict closely to the correct label will no longer be evaluated as performing poorly.

6.2.7 Summary

In this section, an action quality regressor is proposed to fuse Segment-level P3Ds on top of ED-TCN using a regression layer. S3D is a stacked network of multiple

P3Ds trained on local frames. First, it pushes the state-of-the-art on scoring diving and can work even better if having the segment-level labels. It is found that full-video methods learn to focus on water spray. Second, this idea is suitable for more than diving per se. Segment-level fusion [234] is the right direction to pursue given complex actions containing a sequence of actions such as skating and surgery. Third, the optimal version so far is a combination of P3D, ED-TCN and SVR. It is planned to make all jointly trained and tested on scoring figure skating where stage score is available (say, spins). Finally, the performance can be further boosted by ensembling human poses or audience sounds.

The primary insight from this chapter is that semantic segmental 3D networks are very promising to work effectively and efficiently. In the case of diving, it can even be claimed that there is a temporal template that every diving performance fits. S3D should work for tasks where temporal alignment is possible. Figure skating players can design their own sequence of actions: spins earlier or later and different times of spins. It needs careful designs to address the flexible sub-action problem. Techniques such as dynamic time warping should be ready to rescue.

6.3 Conclusion

In this chapter, a novel method is designed for the online learning of priors from Web data. The priors are naturally used in Graph cuts. The Graph cuts algorithm is widely used to formulate image segmentation as an optimization problem, while it does not work well in dynamic scenes. The prior learning method proposed in this work is able to assist Graph cuts effectively to achieve reasonably good segmentation performance in videos with dynamic scenes. The novelty lies in the paradigm of learning priors online:

(1) Online selection of training data: relevant silhouette images for training are online selected using a user-provided bounding box and an object class annotation.

(2) Online learning of priors: a holistic shape energy term is learned for the object, while the object and background seed labels are propagated between frames. This work has been published in our paper [10].

Secondly, the representation of a video is a 3D graph to the deep learning version - spatiotemporal 3D ConvNet. Action quality assessment is crucial in areas of sports, surgery and assembly line where action qualities need to be evaluated. Convolutional neural networks (ConvNets, CNNs) seems a good fit to automate this end-to-end process of estimating a score for a given video. Convolutional 3D (C3D) networks are the type of CNNs that naturally capture both appearance and subtle motion cues in videos. However, training C3D is practically challenging due to memory issues and thus restrictive due to the fixed-sized temporal window of inputting clips.

As a result, a lot of popular practices can be treated as explicitly factorizing C3D into a spatial 2D CNN (say, Inception, Residual Network - ResNet) that trains on images and a temporal 1D CNN (say, Long Short-Term Memory - LSTM, Temporal Convolutional Networks - TCN) that trains on the 1D features generated from the 2D CNN as two steps. This way is unable to capture motion cues until Two-Stream training pattern by additionally training normally the same network on pre-computed optical flow images. In the meanwhile, it is widely known in signal processing theory that a separable filter can be written as a product of more simpler filters. The first contribution of this work is to intuitively justify the correctness of factorizing a 3D ConvNets for processing 3D video volumes as per-layer matrix convolutions of spatial 2D convolutions and temporal 1D convolutions. Indeed, the feasibility and effectiveness of implicitly factorizing 3D convolutions in a unified network have recently been shown empirically in the works of Pseudo-3D (P3D) ResNet and Two-Stream Inflated 3D (I3D) ConvNet that are trained on the large-scale Kinetics dataset for action classification. P3D can be judged as finetuning the pre-trained ResNet with temporal 1D convolutions using 3D video volumes while I3D can be seen as inflating the pre-trained 2D Inception network to 3D along the temporal dimension. Taking P3D as an example choice, our second contribution is to propose the Segment-based P3D-fused network named S3D that stacks multiple P3Ds using fully connected layers for action quality regression. It replaces the fixed-sized clips in previous works with semantic segments given by the Encoder-Decoder TCN (ED-TCN) in a unified

framework. As the third contribution, it is also shown that the temporal segmentation given by ED-TCN can be embedded with few extra efforts and is very accurate. S3D pushes the performance on the UNLV-Dive dataset by a significant margin and performs way better than full-video training. This segment-aware pattern for training 3D CNNs should be readily generalizable to other choices of 3D CNNs such as I3D and actions that contain sub-actions with relatively strong temporal patterns such as surgery. This work is published in our paper [248].

Chapter 7

Supervised Hashing via Learning Embedding and Quantization

A sequence of images usually come with more storage memory as well as more processing and indexing time than an image. Its practical use is highly dependant on the powerful computation of the hardware and efficient algorithm of hashing a set of images. Compared with unsupervised hashing, supervised hashing commonly illustrates better accuracy in many real applications by leveraging semantic (label) information. However, it is tough to solve the supervised learning problem directly because it is essentially a discrete optimization problem. Some other works try to solve the discrete optimization problem directly using binary quadratic programming, but they are typically too complicated and time-consuming while some supervised hashing methods have to solve a relaxed continuous optimization problem by dropping the

discrete constraints. However, these methods typically suffer from poor performance due to the errors caused by the relaxation manner.

As exampled in the first chapter, finding a certain event such as goals in a soccer game video is highly demanded in online video services. Once again, video event retrieval is different from video retrieval, which is just searching a subset of videos out of a large set of videos. However, both problems reply on the techniques of retrieval models and algorithms, and particularly nowadays hashing algorithms. If an image set is used to represent videos, then the feature representation is huge. Hashing comes to play given this context. Then the core problem is to learn a reasonable hash function. It can be a random project as done in Locality-Sensitive Hashing (LSH) or a learned embedding as done in supervised hashing. The supervised hashing is a good choice when there are labels available.

In this section, a new method is proposed to solve the problem introduced by relaxing the cost function. Inspired by the property of rotation invariance of learning embedding features, our method tries to jointly learn similarity-preserving representation and rotation transformation for better quantization alternatively. A generic two-step framework is proposed for supervised hashing: learning binary embedded codes and learning hash functions. The experiments on various datasets show that the proposed method achieve a significant improvement over previous works based on discrete optimization. The proposed methods even achieves the state-of-the-art performance in some image retrieval tasks. There are unsupervised hashing off-the-shelf.

However, when there is label information available, it is wise to choose to use them.

In the following, this section first discusses the learning binary embedding in the general two-step approach [249]. Then it describes the issues of relaxed continuous optimization, which can be interpreted under a probability model. Last, the details of our hashing model are presented, including the model formulation and learning algorithms, and explain the advantage of our method.

7.1 Learning to hash related works

Approximation Nearest Neighbor (ANN) search plays a fundamental role in today big data age. With the explosive growth of data on the Internet, it is difficult to employ traditional Nearest Neighbor (NN) search due to computational cost and network communication. Actually, in many real applications, the exact NN search is unnecessary for every possible query. So it makes us think about leveraging the ANN search method to improve speed and reduce memory capacity. For example, it only needs 4MB memory to store 1 Million points with 32-bit code, and a few milliseconds to search all points. Thus learning to hash is an efficient way for this goal.

There are two main categories of methods for hashing methods: data-independent methods and data-dependent methods. Representative data-independent methods include locality sensitive hashing (LSH) [250, 251] and shift invariant kernel hashing (SIKH) [252]. Compared with the data-dependent approaches, data-independent

ones usually need longer codes to achieve similar performances. Therefore, data-independent methods do not work well in relative low bits quantization. To solve the shortcomings of data independent methods, recent research pays more attentions to data-dependent methods whose hash functions are learned from training data via data-driven methods, which can be further divided into two categories: unsupervised methods [253–256] and supervised hashing [249, 255, 257–260]. And the latter has attracted more and more attention in recent years because it has shown better accuracy than other hashing methods in many real-world tasks. Supervised learning [257, 261, 262] is also able to utilize supervised (label) information to learn the hash codes and thus make it more compact and semantic. Although a couple of learning to hash methods are mainly for unsupervised setting, they can be adapted to supervised setting if the label information is available.

However, it is not easy to solve the problem of learning to hash problem which is essentially a discrete optimization problem. Most existing supervised hashing methods have to solve a relaxed continuous optimization by dropping the discrete constraints. Although the relaxed continuous optimization has some merits such as low computational complexity, these methods typically suffer from poor performances due to the errors caused by the relaxation. Some other methods [262, 263] try to directly solve the discrete optimization problem and achieve the state-of-art performance. However, they are typically time-consuming and unscalable compared with hashing methods using relaxed continuous optimization.

This chapter argues the reason that relaxed continuous optimization may suffer poor performance due to the neglect of the quantization error in the cost function. Without an appropriate cost to penalize quantization error, the solution would be that higher-variance dimensions typically carry more information, but in the quantization process, the same number of bits is used for each dimension, which yields poor performance. To solve the problem, a novel algorithm is proposed to simultaneously take both quantization and distance approximation into account. It is able to solve the problem of performance loss caused by using the relaxed continuous optimization. The experiments are conducted on three different datasets with semantic labels, it is demonstrated that our method is competitive compared with other state-of-the-art methods based on discrete optimization especially in the dataset with large categories.

7.2 Problem Definition

Suppose there are n images $X = \{x_i\}_{i=1}^{n}$ where x_i is the feature vector of point i. x_i can be the hand-crafted features or the raw images in a retrieval problem. Besides the feature vectors, the training set also contains a set of pairwise labels $S = \{s_{ij}\}$ with $s_{ij} \in \{1, -1\}$, where $s_{ij} = 1$ if x_i and x_j belong to the same class or $s_{ij} = -1$ otherwise.

The supervised hashing with pairwise labels is to learn a binary code $b_i \in \{-1, 1\}^c$ for each point x_i, where c is the code length. Instances with similar labels should

have similar binary codes $B = \{b_i\}_{i=1}^n$ which preserve the similarity in S. More specifically, if $s_{ij} = 1$, the Hamming distance between binary codes b_i and b_j should be low. Otherwise if $s_{ij} = -1$, the hamming distance between binary codes b_i and b_j should be high. The goal of supervised hashing is to learn a binary code matrix $B = [b_1, ..., b_i, b_j, ..., b_n]$ and then a function can be learned for the binary code as $b_i = h(x_i) = [h_1(x_i), h_2(x_i), ..., h_c(x_i)]^T$, where $h(x_i)$ is the hash function to learn.

7.3 Supervised hashing with pairwise labels

In most of the existing supervised hashing methods, the side information is in the form of pairwise labels that indicate the semantic similarities or dissimilarities on image pairs. The loss functions in these methods are thus designed to preserve the pairwise similarities of images. Let $\theta_{ij} \in \Theta$ denote half of the inner product between two binary codes $b_i, b_j \in \{-1, 1\}^c$, where

$$\theta_{ij} = \frac{1}{2} b_i^T b_j, \tag{7.1}$$

and the likelihood of the observed similarity labels S can be defined as follows

$$P(S|B) = \prod_{s_{ij} \in S} p(s_{ij}|\sigma(\theta_{ij})). \tag{7.2}$$

It is easy to see that different $P(S|B)$ can be obtained by using different $\sigma(\theta_{ij})$ functions with different prior probability. If the model defines $\sigma(\theta_{ij}) = \frac{\theta_{ij}}{c}$ and assumes

that the $P(S_{ij}|\sigma(\theta_{ij}))$ has normal distribution conditional on the binary codes B, then maximizing the likelihood of S in Eq. 7.2 would be equivalent to optimize the following objective function [249, 260, 262, 263]:

$$\begin{aligned} &\max_{B\in\{-1,1\}^{N\times c}} \sum_{s_{ij}\in S} \log P(s_{ij}|\sigma(\theta_{ij})) \\ =&\min_{B\in\{-1,1\}^{N\times c}} \sum_{s_{ij}\in S} (s_{ij}-\theta_{ij})^2 \end{aligned} \tag{7.3}$$

The KSH loss function is based on hamming affinity using L_2 loss function, which is also used in MDSH [264]. Alternatively, if defining $P(s_{ij}|\sigma(\theta_{ij}))$ with

$$P(s_{ij}|\sigma(\theta_{ij})) = \frac{1}{1+e^{-s_{ij}\theta_{ij}}} \tag{7.4}$$

then maximizing the likelihood of S in Eq. 7.2 would be equivalent to optimize the following objective function:

$$\begin{aligned} &\max_{B\in\{-1,1\}^{N\times c}} \sum_{s_{ij}\in S} \log P(s_{ij}|\sigma(\theta_{ij})) \\ =&\min_{B\in\{-1,1\}^{N\times c}} \sum_{s_{ij}\in S} \log(1+e^{-s_{ij}\theta_{ij}}) \end{aligned} \tag{7.5}$$

It is obvious that these both two object functions are essentially a discrete optimization problem which is hard to optimize. Therefore, some algorithms try to relax $B \in \{-1, 1\}^{N\times c}$ to be a real-valued matrix $U \in \mathbb{R}^{N\times c}$, then the $p(S|B)$ can be turned to $P(S|U)$ and the $\theta_{ij} = \frac{1}{2}u_i u_j^T, u \in R^c$. The optimal B is obtained through Single-Bit Quantization (SBQ), where the real values of each dimension are quantized to $\{-1, 1\}$ by thresholding.

However, there is a very important problem: higher-variance dimension carry more information, using the same number of bits for each dimension yields poor

performance. In LFH [261], the Gaussian prior is $P(U) = \prod_{i=1}^{c} N(u_{*j}|0, \beta\mathbf{I})$ (u_{*j} is j-th dimension of U), but it cannot be used to solve the problem efficiently. Instead, $P(B|S)$ is extended as the following equation,

$$P(B|S) = P(S|U)P(U|B). \tag{7.6}$$

This chapter proposes a new method which can solve the problem more efficiently by balancing the embedding results with real value and quantization error. Please note that only the objective function Eq. 7.5 is considered due to space limitation. The extension of the proposed model such as Eq. 7.3 is beyond the scope of this dissertation.

7.4 Jointly Learning Embedding and Quantization (JLEQ)

Basically, quantization is turning real-valued features into binary codes. Using Iterative Quantization [255] is a direct way for unsupervised hashing methods but only a few works discuss its application in supervised hashing method (CCA-ITQ in [265]). So far as is known, no works discuss the application of using ITQ in hashing methods based on embedding approaches. In the empirical analysis, it is found that the ITQ cannot improve the performance of two-stepped methods as simple as the CCA-ITQ. Thus based on the property of rotation invariance of embedding techniques, a novel

way is proposed to jointly learn representation and quantization based on the two-stepped framework [249]. More specifically, it can be defined that the $P(B|U)$ in Eq.7.6 should be as the following:

$$\begin{aligned} P(B|U) &= \prod_{i=1}^{n} P(b_i \mid u_i; R) \\ &= \prod_{i=1}^{n}\prod_{j=1}^{c} \frac{1}{\sqrt{2\sigma^2\pi}}\, e^{-\frac{(b_{ij}-R_j u_{ij})^2}{2\sigma^2}} \end{aligned} \tag{7.7}$$

and then the negative log posterior of $P(B|S)$ can then be derived as:

$$\begin{aligned} L &= -logP(B|S) \\ &= \sum \log(1+e^{-s_{ij}\theta_{ij}}) + \sum_{i=1}^{n} \|sign(u_i) - Ru_i\|_2^2 \end{aligned} \tag{7.8}$$

the penal term $\|sign(u_i) - Ru_i\|_2^2$ is able to control the space distribution of u_i without changing distance. So for the embedding problem in Eq. 7.5, the smaller the quantization loss such as $\|sign(u_i) - Ru_i\|^2$, the better the result will be an approximation of binary codes. So it worthies calling the Eq.7.6 using Eq.7.7 as JLEQ (Jointly Learning Embedding and Quantization). Therefore, U and R can be minimized in an alternating procedure. Please note this procedure need many iterations.

Fixing U, optimize R. This reduces to the Orthogonal Procrustes Problem (OPP):

$$\sum_{i=1}^{n} \|sign(u_i) - Ru_i\|_2^2 \;\; s.t. R^T R = I_c \tag{7.9}$$

where an optimum of OPP can be obtained as same as the method in [265], and then U is updated as RU.

Fixing R, optimize U. Since directly optimizing the whole B and U is impossible when the dataset is too large because it would be very time-consuming, so a simple

strategy more feasible is to optimize each row of U at a time with its other rows fixed. The second order stochastic gradient descent algorithm [261, 266] can be used to optimize each row $u_i \in U$. The gradient vector and the Hessian matrix of the object function $-\log(P(S|U))$ defined in Eq. 7.5 can be derived as:

$$\frac{\partial L}{\partial u_i^T} = \sum_{i=1}^{n}\sum_{j=1}^{n} -s_{ij}\sigma(-s_{ij}\theta_{ij})u_j^T \tag{7.10}$$

$$\frac{\partial^2 L}{\partial u_i^T \partial u_i} = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \sigma(-s_{ij}\theta_{ij})(1-\sigma(-s_{ij}\theta_{ij}))u_j^T u_j \tag{7.11}$$

where $\sigma(-s_{ij}\theta_{ij})(1-\sigma(-s_{ij}) \leq \frac{1}{4}$, the and the thus hessian matrix $\mathbf{H}$ can be defined as [267]:

$$H = \frac{1}{8}\sum_{j=1}^{n} u_j^T u_j \tag{7.12}$$

7.5 Out-of-sample Extension

The above section has presented the detail of our method to learn a binary code. However, for new query u not in the training set, the binary code b still needs to be computed. Many supervised hashing methods based on the two-step strategy in [249] which first learns binary code and then trains a linear transformation from the original input to the binary code. Because of leveraging this strategy, our method like [261–263] still need to train a matrix $W \in R^{D\times C}$ that maps x to u in the following ways:

$$u = W^T x \tag{7.13}$$

Then the binary code can be easily obtained by $sign(u)$. Commonly, the W can be learned using linear regression with regularization like:

$$\min_{W} \|U - XW\|_F^2 + \lambda \|W\|_F^2 \tag{7.14}$$

where $U = [u_1, ..., u_n]$ is the embedding result of the original input X. The optimal W can be represented as a closed form:

$$W = (X^T X + \lambda I)^{-1} X^T U \tag{7.15}$$

where I is an identity matrix.

7.6 Experiments

In the following, the experimental results are presented. Our model is compared with several baselines on three widely used benchmark datasets: CIFAR-10, CIFAR-100 [268], and NUS-WIDE [269].

7.6.1 Datasets

The CIFAR-10 dataset consists of 60,000 color images (32×32) which are categorized into ten classes (6000 images per class). It is a single-label dataset in which each image belongs to one of the ten categories. The CIFAR-100 dataset is just like the CIFAR-10, except it has 100 classes containing 600 images each. The NUS-WIDE dataset has nearly 270,000 images collected from the Internet. It is a multi-label

dataset in which each image is annotated with one or multiple class labels from 81 classes. Following [260], there also exist some images without any label, which are not suitable for our evaluation. After removing those images without any label, 209,347 images are obtained for our experiment. Two images are considered to be semantically similar if they share at least one common label. Otherwise, they are semantically dissimilar.

To be independent of deep feature's representation power, this work has not touched upon deep features. Instead, conventional hand-crafted features such as GIST are used for all hashing methods during evaluation. Each image in CIFAR-10 and CIFAR-100 is represented by a 512-dimensional GIST vector. Each image in NUS-WIDE is represented by an 1134 dimensional low-level feature vector, including 64-D color histogram, 144-D color correlogram, 73-D edge direction histogram, 128-D wavelet texture, 225-D block-wise color moments and 500-D bag of words based on SIFT descriptions.

7.6.2 Settings

Experiments in [261, 263] showed that supervised methods can outperform unsupervised methods, thus our method is only compared with several state-of-the-art supervised hashing methods including SPLH [258], KSH [260], TSH [249], FastH [263], LFH [261], SDH [257], COSDISH [262] and CCA-ITQ [255, 265]. All the baselines are implemented using the source code provided by the corresponding authors. For LFH

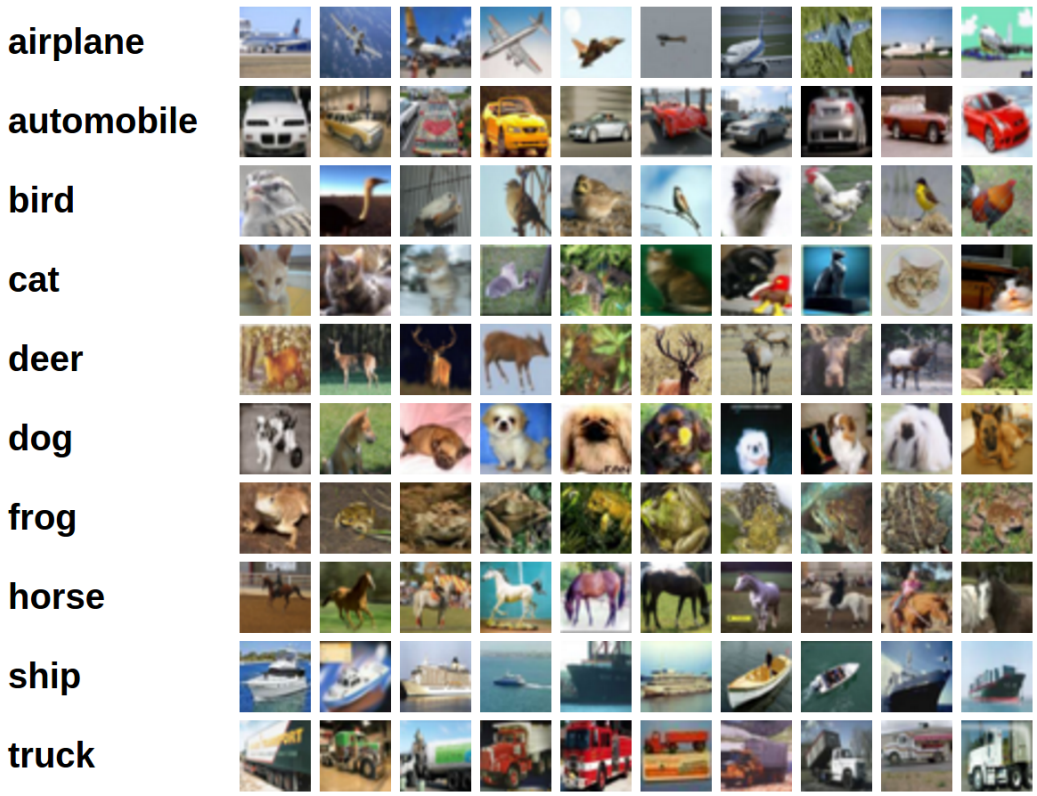


Figure 7.1: Cifar 10 dataset. Given a query, returning the pictures or videos. Retrieval is slightly different from recognition: given a truck query image, find other truck images in the gallery with a similarity ordering.

and COSDISH, the model uses the stochastic learning version with 50 iterations and each iteration samples q columns of the semantic similarity matrix as in [261, 262]. For SPLH, KSH and TSH, using the entire training set for training is impractical due to high time complexity. Same with [260], the model randomly samples 2000 points as the training set for CIFAR-10 and 5000 points for NUS-WIDE. TSH [249] can use different loss functions for training. To be fair, the loss function in KSH is used for training TSH and the SVM with RBF-kernel is used for out-of-sample-extension in TSH. For KSH, the number of support vectors is 300 for CIFAR-10, and 1000 for

Table 7.1: Results of MAP on CIFAR 10

Method	8-bits	16-bits	32-bits	64-bits
Ours (JLEQ)	0.5153	0.5811	0.6147	0.6353
LFH	0.2908	0.4098	0.5446	0.6038
SDH	0.2642	0.3994	0.4145	0.4346
TSH	0.2365	0.3080	0.3455	0.3663
KSH	0.2334	0.2662	0.2923	0.3128
SPLH	0.1588	0.1635	0.1701	0.1730
COSDISH	0.4986	0.5768	0.6191	0.6371
FastH	0.4230	0.5216	0.5970	0.6446

NUS-WIDE. For FastH, boosted decision trees are used for out-of-sample extension rather than linear regression. The entire training set is used for FastH training on CIFAR-10, and randomly sample 100,000 points for FastH training on NUS-WIDE. As with most existing hashing methods, the mean average precision (MAP) is used to measure the accuracy of our proposed method and other baselines.

7.6.3 Results

The mean average precision (MAP) is a widely used metric for evaluating the accuracy of hashing. Table 1, table 2 and table 3 show the MAP of our method and baselines with different code lengths on the three datasets, respectively. In table 1, compared with other methods using relaxed constraints to optimize hash functions,

Table 7.2: Results of MAP on NUS-WIDE

Method	8-bits	16-bits	32-bits	64-bits
Ours (JLEQ)	0.5892	0.6134	0.6023	0.5940
SDH	0.4739	0.4674	0.4908	0.4944
LFH	0.5437	0.5929	0.6025	0.6136
TSH	0.4593	0.4784	0.4857	0.4955
KSH	0.4275	0.4546	0.4645	0.4688
SPLH	0.3769	0.4077	0.4147	0.4071
COSDISH	0.5454	0.5940	0.6218	0.6329
FastH	0.5014	0.5296	0.5541	0.5736

Table 7.3: Results of MAP on CIFAR 100

Method	8-bits	16-bits	32-bits	64-bits
Ours (JLEQ)	0.6029	0.6055	0.6298	0.6268
LFH	0.6007	0.6012	0.6086	0.6009
COSDISH	0.5638	0.5564	0.5826	0.5818
CCA-ITQ	0.5053	0.5039	0.5076	0.5080

Table 7.4: Results of MAP on CIFAR 100

Method	8-bits	16-bits	32-bits	64-bits
Ours (JLEQ)	0.6029	0.6055	0.6298	0.6268
LFH	0.6007	0.6012	0.6086	0.6009
COSDISH	0.5638	0.5564	0.5826	0.5818
CCA-ITQ	0.5053	0.5039	0.5076	0.5080

our method outperforms all of them in CIFAR-10. Although the gap between the proposed method and FastH is 64-bits, our method still has a significant advantage in the setting of shorter code lengths. However, the FastH baseline exploits random forest rather than linear regression as the classifier that commonly works better than linear regression in this cases. In Table 2, our method gets the pretty good performance of 8-bits and 16 bits settings in NUS-WIDE and it is even superior to the methods based on discrete optimization. Although it is still not as good as COSDISH in 32, 64-bits. Compared with CIFAR-10, CIFAR-100 has more classes (100 categories) and fewer samples for each category. Thus it is a large challenge for hashing methods. As shown in Table 3, as the code length increased, the performance of the proposed method consistently increases, but other methods just have slightly perturbation in different code lengths. Only three methods are selected in this dataset because the LFH is the state-of-arts of hashing method using relaxed optimizations. COSDISH works better than FastH.

7.7 Summary

This chapter proposes a novel strategy of jointly learning embedding and quantization for supervised hashing. It only needs relaxed continuous optimization so that it is easy to be solved without sophisticated optimization methods. Hence, it is also easy to handle large-scale problems. Experimental results on three datasets with semantic labels show that our methods achieve competitive performance compared with the state-of-the-art. In this work, the model is built based on the general two-step framework - learning binary embedded codes and learning hash functions, which is generic enough to be useful for application such as video retrieval and even video event retrieval when there is a good coding of events. A new method is proposed to solve the problem introduced by relaxing the cost function. Inspired by the property of rotation invariance of learning embedding features, our method tries to jointly learn similarity-preserving representation and rotation transformation for better quantization alternatively. In experiments, our method shows significant improvement. Compared with the methods based on discrete optimization, our method obtains the competitive performance and even achieves the state-of-the-art performance in some image retrieval applications. This work has been published in our paper [270].

Chapter 8

Conclusion

Albert Einstein said, everything should be made as simple as possible, but not simpler. As complicating things is straightforward, I am very much curious if a certain simple model really does not work well. This research philosophy also drives the work of this dissertation. In the following, I first summarize the all the presented works and emphasize again the contributions again. Then, I touch upon some future research work which could potentially make the dissertation even more coherent from various perspectives and some long-term future work which could lead to another dissertation along this research direction. In the end, I give a couple of examples as regards how the presented research work can be applied in real work such as industrial applications.

8.1 Contributions

Image Sequences come in various forms. For example, a video is a sequence of images frame by frame over time; a 3D volumetric image is a sequence of images slice by slice over one dimension of the 3D space [ISBI'18a]; a typical multiresolution image analysis is sliding over and convolving an image with different kernel size to induce a sequence of image patches over the scale space [ISBI'18b].

Now, if a video is simply modeled as as a set of permutable images, the representation lose the sequential information. Possibly due to the obvious drawback, this area is not heavily investigated in the recent literature. In fact, the model can still learn an informative representation from an image set. One approach of relatively high popularity nowadays is pooling or aggregating a single feature vector [ICPR'16w1]. Another approach from a more intuitive perspective is exploiting the low-rankness over image intensities and group sparsity over the linear representation vectors so that all the images collaboratively represent a set of images [ICASSP'15, TCSVT'18]. The elegance of the model is also given by the fact that linear is simple.

On the other hand, sequential modeling is a well-studied area: global *vs.* local, long-term *vs.* short-term [ICPR'16 w2], Markov *vs.* semi-Markov *vs* autoregressive [ACCV'12], and so on. However, I tend to revisit simple models such as frame/slice differentiating, phase discrepancy, motion modeling [ACCV'12, MICCAI'14w], and correlating features over time.

With such a "less is more" mindset, I even doubt if going from an image to the

image sequence is always necessary. At least in certain real-world computer vision systems, it is critical to making recognition, prediction and decision based on the visual information of exactly here right now. In that case, it is wise to make a full use and a deep exploration of an image. One empirical work I performed is to transfer image features for a different task simply by regularizing the image-based model [ICIP'17a]. Another example is gaining the discriminative power by normalizing the deep features as well as the weights of deep networks.

A sequence of images usually come with more storage memory as well as more processing and indexing time than an image. Its practical use is highly dependant on the powerful computation of the hardware and efficient algorithm of hashing a set of images [ICIP'17b]. While I am not sure if what Einstein says is always correct, I have already realized the beauty of a simpler model - less computation, less memory occupation, less parameter tuning, or simply better understanding. In the case when computation, memory and less tuning are important to us, I may want to simply maximize the performance of a simple model [ACM MM'17].

First, this dissertation has extended the models of representing face videos as a set of deep features (see Chapter 3). it has proposed an elegant linear model to untangle facial actions from expressive face videos which contain a mixture of linearly representable attributes. In addition, it has also proposed a network that fine-tunes a state-of-the-art face verification network using a regularized regression loss and additional data with expression labels.

Second, it has also extended the model of representing a face video using inter-frame models (see Chapter 4). It has proposed an elegant linear model to untangle facial actions from expressive face videos which contain a mixture of linearly representable attributes. Moreover, it has formulated expression recognition as a video Sparse Representation based Classification (SRC) with Long Short-Term Memory (LSTM) mechanism.

Third, this dissertation has looked the temporally local representation of egocentric videos using motion models (see also Chapter 5). The models can be between adjacent frames such as global motion estimation, or among a few of frames with the assumption of Markov properties such as Bayesian filtering. It has investigated how sensitive the estimated motion is affected by the number of motion models assumed in feature matching. It has also designed a robust tracker called MIL-PF combining the merits of the Multiple Instance Learning tracker and Particle Filter tracker.

Fourth, it has made two novel designs for spatiotemporal representation of human action videos as 3D graphs or 3D ConvNets (see Chapter 6). We have designed an object-class independent framework to segment moving objects in videos with dynamic scenes, say, moving persons. Furthermore, we have also proposed a human action quality regressor S3D that fuses Segment-level P3Ds on top of ED-TCN using a regression layer.

Lastly, motivated by finding a certain event for online video services and th great application progress of supervised learning, this dissertation has proposed a novel

strategy of jointly learning embedding and quantization for supervised hashing. Although video event retrieval is different from video retrieval, both problems reply on the technique of retrieval models and algorithms, and particularly nowadays hashing algorithms.

In summary, this dissertation examines three ways to model facial or human actions from videos: image-set, temporal and spatiotemporal modeling. Tasks range from action recognition to action quality assessment. While the contribution of each chapter has been listed in the Conclusion section of that chapter, the primary contributions of the whole dissertation can be categorized as followed.

(1) It is verified through several proposed models that image set works effectively for facial actions that are more about summarization. The collective representation is given by (i) collaborative modeling and the efficiency is given by (ii) selective aggregation.

(2) It is shown that weakly supervised on-line manner does not necessarily result in better robustness, but does enable recovering from the error. In addition, it is a good choice to maintain an off-line classifier together with the on-line classifier. Moreover, Particle filter tracker cannot track the target very accurately but does not drift far away. However, spatiotemporal representations are needed to model human actions which are more complex both in space and time. As a result, I also propose

(3) a set of models are proposed to localize, track, segment and recognize an action . Unlike two streams, motion cues are used in a unified framework such as (a) 2D

spatial and 1D temporal factorized 3D CNN, (b) path optimization on a 3D graph and (c) naturally strengthening appearance based tracker with a motion model by equating the sample in particle filtering with the sample in multiple instance learning.

The state-of-the-art performances have been achieved for tasks such as quantifying actions of the facial pain and human diving.Other conclusions of this dissertation are categorized as follows: (i) The sparse and low-rank representations of facial actions can have the expression, identity and poses cues untangled and can be learned using an image-set model and also a linear model; (ii) It is shown from face nets that norm is related with recognizability and that the similarity metric and loss function matter; (iii) Segmenting object locally makes it amenable to assign shape priors and also it is feasible to learn knowledge such as shape priors online from the rich Web data with weak supervision; (iv) It works locally in both space and time to represent videos as 3D graphs and also 3D CNNs work effectively when inputted with temporally meaningful clips.

In terms of practical significance, it is hoped that the proposed models will lead to working components in building face and gesture recognition systems. In addition, the models proposed for videos can be adapted to other modalities of sequential images such as volumetric medical images which are not included in this dissertation. A list of all publication can be found at `https://scholar.google.com/citations?user=-D5k5ioAAAAJ&hl=en`. The key publication that are referred to preivously in this section are listed in the following.

Section reference.

[ICPR'16w1] Xiang Xiang and Trac D. Tran: Pose-Selective Max Pooling for Measuring Similarity. FFER workshop at IAPR ICPR 2016, Cancun, Mexico.

[ICASSP15] Xiang Xiang, Minh Dao, Gregory D. Hager, Trac D. Tran: Hierarchical Sparse and Collaborative Low-Rank Representation for Emotion Recognition. IEEE ICASSP 2015, Brisbane, Australia.

[TCSVT18] Xiang Xiang and Trac D. Tran: Linear Disentangled Representation Learning for Facial Actions. IEEE Trans. Cir. Sys. for Video Tech. (T-CSVT), vol. PP, iss. 99, 2018.

[ICPR'16w2] Xiang Xiang and Trac D. Tran: Recursively Measured Action Units. MPRSS at ICPR 2016, Cancun, Mexico.

[ACCV'12] Xiang Xiang, Hong Chang, Jiebo Luo: Online Web-Data-Driven Segmentation of Selected Moving Objects in Videos. ACCV 2012: 134-146, Daejeon, Korean.

[MICCAI'14w] Xiang Xiang, Daniel Mirota, Austin Reiter, Gregory D. Hager: Is Multi-Model Feature Matching Better for Endoscopic Motion Estimation? CARE workshop at MICCAI 2014, Boston, USA.

[ICIP'17a] Feng Wang, Xiang Xiang, Chang Liu, Trac D. Tran, Austin Reiter, Gregory D. Hager, Harry Quon, Jian Cheng and Alan L. Yuille:Regularizing Face Verification Nets For Pain Intensity Regression. IEEE ICIP 2017, Beijing, China.

[ICIP'17b] Hao Zhu, Feng Wang, Xiang Xiang and Trac D. Tran. Supervised Hash-

ing with Jointly Learning Embedding and Quantization. IEEE ICIP 2017, Beijing, China.

[ACM MM'17] Feng Wang, Xiang Xiang, Jian Cheng and Alan L. Yuille: NormFace: L2 Hypersphere Embedding for Face Verification. ACM MultiMedia 2017, Mountain View, USA.

[ISBI18b] Wentao Zhu, Xiang Xiang, Trac D. Tran, Gregory D. Hager and Xiaohui Xie. Adversarial Deep Structural Nets for Mammographic Mass Segmentation. IEEE ISBI 2018.

[ISBI'18a] Xiang Xiang, Trac Tran, Greg Hager. Multi-Scale CNNs for Lung Nodule Detection. IEEE ISBI 2018, Washington DC, USA.

[ICIP'18] Xiang Xiang, Ye Tian, Austin Reiter, Gregory D. Hager, Trac D. Tran. S3D: Stacking Segment-level P3D for Action Quality Assessment. To appear at IEEE ICIP 2018, Athens, Greece.

8.2 Future works

Till now, I assume there exists a certain action given a video so that the proposed models can be used to spatially localize the action, model the action and recognize the action. In real applications, however, a system needs to detect when a certain action happens, namely temporally localizing the action. As shown in Fig. 8.1, a movie contains numerous scenes. It requires a much deeper semantic understanding

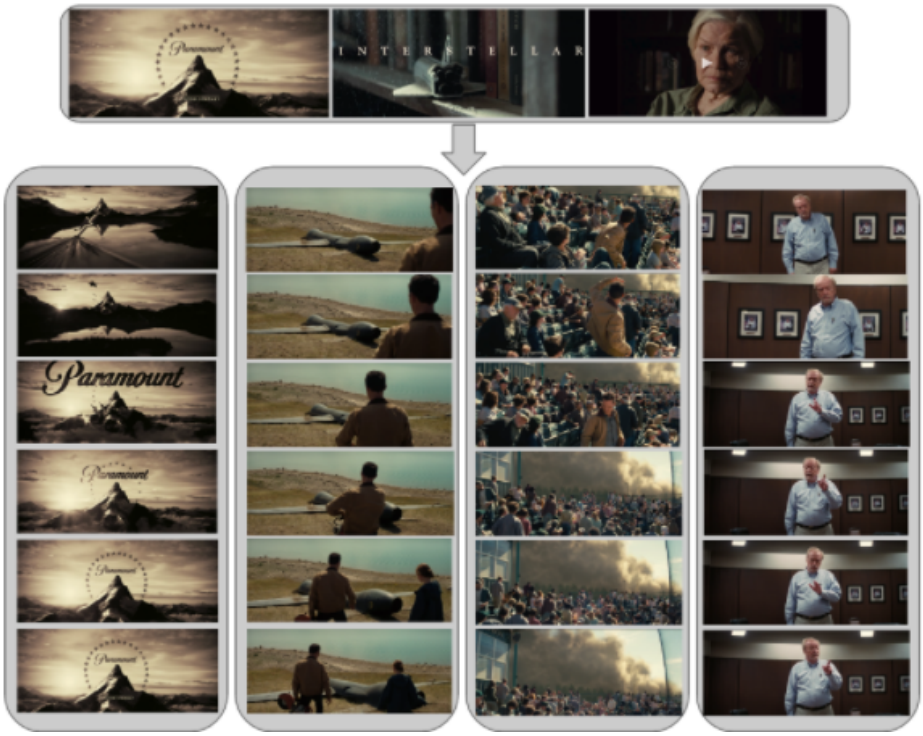


Figure 8.1: The basic idea of clustering movie frames visually into scenes. Shown using the movie Interstellar (2014).

of the video for a model to be able to cluster frames into scenes accurately. Similarly, detecting keyframes, changes and particular events in videos are highly needed in a working system (see Fig. 8.2 and Fig. 8.3).

Furthermore, the representation of objects such as the face and human body helps computer programs precept them in the real world. Observing that the same object (input space) has various attributes (output space), I've been interested in robust mappings with attribute disentanglement in real-world visual data such as videos of faces. Identity-labeled data are rich while quantified expression labels are not. Can we transfer a well-trained face recognizer to expression? Yes! In the linear function

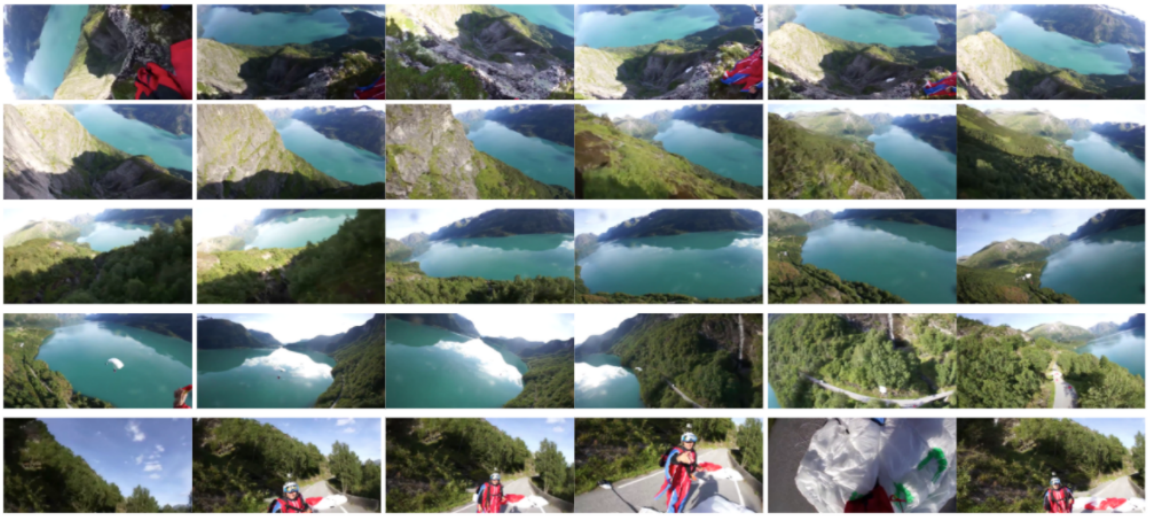


Figure 8.2: Key frames extracted from a video collected using a head-mounted GoPro camera in the SumMe dataset [8].

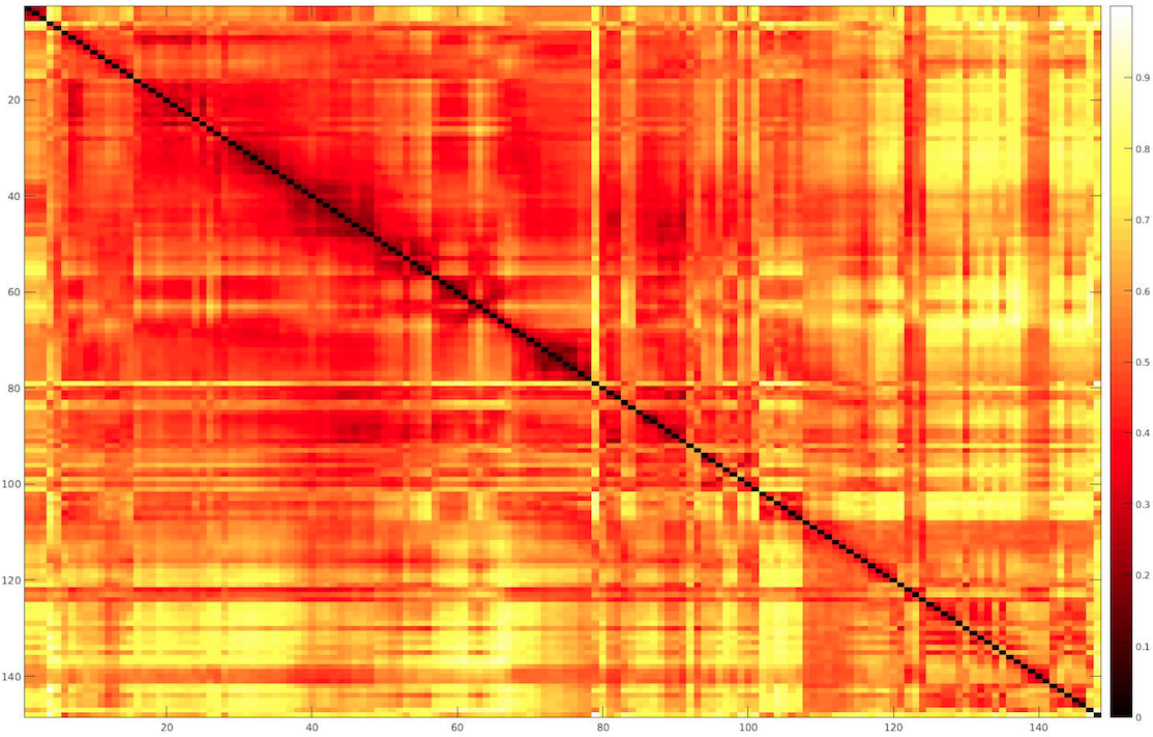


Figure 8.3: Distance matrix in terms of frame energy difference for the above video. There are 148 frames. Thus, the distance matrix is indexed by 1∼148 at both dimension. It is symmetric and the diagonal is supposed to be all 0. Adjacent frames normally are similar and thus there exist dark blocks close to the diagonal.

space, one observation is that a specific expressive face can be a linear combination of various people's faces with the same expression. In the non-linear space, I show how to design a deep network to learn an expression recognizer by refining a deep face recognizer with a few additional training examples with expression labels. These research findings are generalizable to other couples of attributes such as identity and pose. Similarly, a specific person's face at a specific pose can be represented using a bag of the same person's faces at various poses. Beyond discriminative models, the learned knowledge relating attributes can serve as the prior knowledge to "drive" the object to "move" from a static image in novel ways, namely to generate instances of novel attributes such as synthesis of a novel view or expression. The methodology discussed above should be applicable to other deformable objects such as the human body and robots considering the kinematics of objects.

In addtion, the temporal and spatiotemporal reasoning is a fine-grained extension of the current action recognition works.

8.3 Potential applications

This section gives examples of today's surveillance cameras that are slightly different from our impression of public security surveillance camera shown in Fig. 8.4.

Home-camera surveillance videos. Very recently, my mother's bag handbag left on the sofa is stolen at home when the front door is open while she is in the kitchen. When my mother calls my father and also the police, they instantly turn to the surveillance camera. My father has installed a PTZ camera on the top of the wall where the sofa is attached. Now I can watch the monitoring video through a phone application (app.).

Fortunately, we manually fast forward the past two hours but have not seen anyone entering the living room except my mother (see Fig. 8.5). It turns out that the theft is so crafty that he or she may slip in closely attaching to the wall to stay in the camera's dead angle. It is disappointed that the surveillance camera is not able to capture the stealing. If there is another camera on the opposite wall, the theft should be almost for sure captured. There are so many consumer surveillance cameras being used nowadays but not all of them are properly used.

Earth-camera surveillance videos. As shown in Fig. 8.6, consumer surveillance cameras are so pervasive nowadays that a lot of issues which do not exist before are being raised. For example, real-time video communication is reliable now; remote control of the camera is entirely feasible now; privacy and security issues are well explored. The so-called smart cameras nowadays are still not smart enough. Although there exists

Figure 8.4: Screenshots of typical public security surveillance videos. Top left: public security monitoring room with lots of monitors that show the stations, road and so on. Top right: monitoring a parking lot from the TRECVID 2018 challenge. Bottom left: a theft is stealing a suitcase on the luggage rack. Bottom right: a road surveillance camera monitors a robbery that a man is robbing the car next to his car when waiting for the green light. Except for the TRECVID videos, pictures from the Internet.

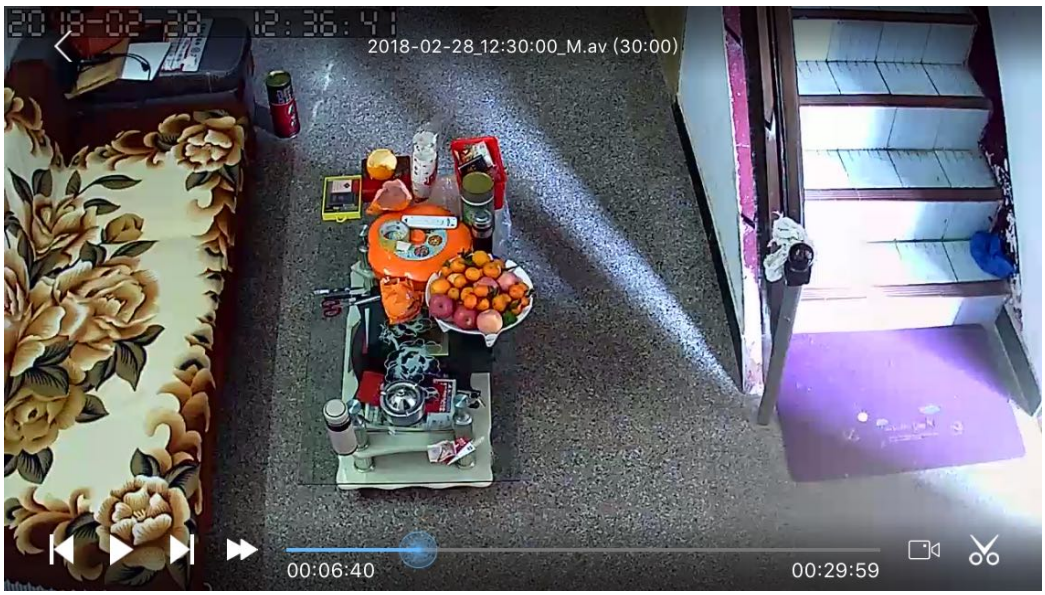


Figure 8.5: A screen-shot of the video captured using a home security camera.

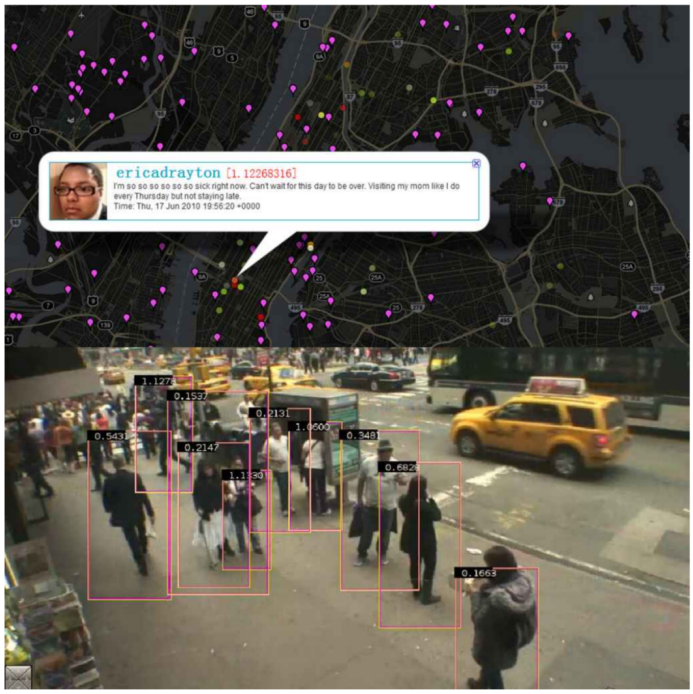


Figure 8.6: Locating a sick ©Tweeter user [9] from ©GPS and ©Earthcam, which is a platform containing ubiquitous webcams that are shared over the Internet.

movement detection and person detection modes, the camera does not distinguish if the person is a family member or a stranger. Many users cannot bear the false alarms whenever there is a person detected. As a result, users turn those modes off. And then when a theft comes, there may be no alarm.

Home care videos. Different from healthcare, homecare is in everyday's life. Household appliance industry such as ©Philips and ©Midea has developed products for the elderly fall detection and infant sleep monitoring using low-resolution cameras for privacy [271]. For example, when my wife wants to let our daughter sleep in her own bedroom, she installs an baby sleep monitoring camera as shown in Fig. 8.7. Whenever she wakes up, there will be an alert sent to our phones. At midnight, we can also

Figure 8.7: Screen-shots of infant sleep monitoring video from a home care camera. Left: the captured bed at night with the light off. Right: the color-enhanced image.

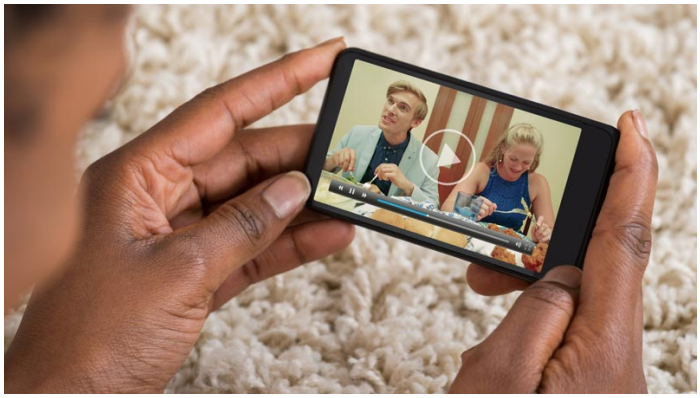


Figure 8.8: Smartphones are ubiquitous and revolutionize how people watch videos.

watch her on the phone app in the colorized mode. Once making sure that she falls back to sleep again, we fell rest assured without bothering to get up to check. The camera seems smart in terms of colorizing videos and detecting the baby's waking up. But I suspect that the solution is probably just detecting whether the baby's eyes open or not. It turns out that it is even simpler. In the end, the alert mode is turned off, as it is too bothering to receive the alarms whenever the baby moves.

Live videos and micro-videos. Just like microblogs to newspapers and micro-

Figure 8.9: Live video scenarios and a screenshot of a phone app. The left is a typical scenario where the lady is performing while a mounted smartphone lively broadcasts this performance. The right is a screenshot of typical live videos where the remote audiences interact with the host or so-called anchor through barrages which are just text all over the screen.

films to studio-produced films, magazines and portal websites, the traditional TV broadcast has been challenged by Internet video broadcast from phone cameras and web-cams supported by applications such as ©Youtube live and ©Facebook live. Nowadays it turns out that smart TVs have become a big thing, which is more like a watcher-friendly computer monitor with Internet access and a portable computer such as ©Apple TV, ©Google TV and ©Mi TV. However, as shown in Fig. 8.8, neither smart TV nor computers can be as convenient as smart-phones which people bring along with everywhere. In this sense, smart-phones are indeed ubiquitous computing device which revolutionizes a lot of application scenarios of video analysis.

Particularly, live videos and micro-videos get popular as a new form of the so-called We media soon after being introduced to social media. With smart-phone

Figure 8.10: Screenshot of a video on the video sharing platform ©Bilibili. The so-called barrages are a lot of on-going or historic chatting texts that are flying all over the screen. However, from monitoring perspective, these barrages become challenging noises occluding and distracting the scene.

apps, technically everyone is free to make and post a micro-video which is potentially able to reach to the world. Indeed, (live) video sharing platform such as ©Youtube, ©Bilibili, ©Tik Tok, ©Kwai and ©Douyu give chances for the unknowns to get paid by fans and thus the video popularity. Some even become Web stars, among whom Justin Bieber and Guo Meimei are well known nowadays, as shown in Fig. 8.9. However, inappropriate contents might be introduced in order to attract viewers.

Live videos or called live streams generalizes the private video chat technique into public sharing websites or phone apps. Now the watchers can interact with the host via chats and the so-called barrages as shown in Fig. 8.10. Nowadays, micro-video lasting less than a song's time become a popular news and entertainment format.

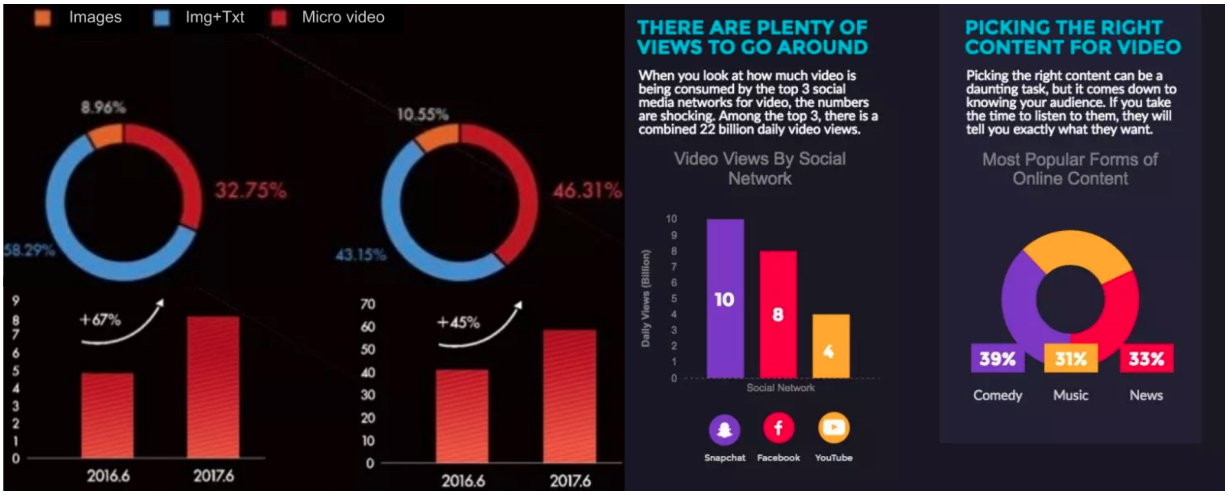


Figure 8.11: Micro-video statistics from the Internet.

Micro-videos or called the instant videos get the name from the fact that unlike live videos they usually last no more than five minutes. Even simply transiting pictures with a proper background music turns out to be more friendly to users than letting them browsing a picture after another. As a result, the music type of micro-videos is also called instant music video. Most of micro-videos nowadays are generated using smart-phone apps. It is easy to edit phone-captured videos.

Figure 8.11 shows a set of statistics about the micro-video market share and content type. The left statics are from ©Toutiao, the No. 1 popular news sharing phone app in China by 2017. The top two implies that the percentage of news in the form of micro-videos have grown from 32.75% in the first half year of 2016 to 46.31% in 2017's first half. Moreover, Micro-videos have become the No.1 popular news format in Toutiao in the first half year of 2016. The bottom left shows that the daily usage

Figure 8.12: Screen-shots of ©iPhone FaceTime (left) and ©Skype (right) video chats. In such applications, recognizing emotion and activities improves the user experience.

times of Toutiao's micro-video platform has increased by 67%. The bottom right shows that the daily average usage hours on its micro-video platform has increased by 45%. The right statistics are from the Internet. ©Youtube is no longer the No.1 video generator. Instead, social media such as ©Snapchat and ©Facebook generates way more videos per day. In terms of contents, videos shared on the Internet are roughly among one of three categories - comedy, news and music. Given the fierce competition, it becomes tricky to pick up the right content to be presented in the right platform. It is high expected that this picking process can be optimized by computer algorithms.

Live-chat and egocentric videos and others. Such video streams may also be egocentric videos captured by dash cams, cameras mounted on autonomous cars, web-cams, phone cameras (see Fig. 8.12) and drone cameras (during aerial video

Figure 8.13: Screen-shots of a soccer game live streamed using © YouTube on a laptop connected with a video camera set up on the stands. Normally, no player is willing to operate the camera personally. We can set up the camera at the high stand so that it captures the while field. But then the players in the video will be too tiny to be interesting enough for viewers to keep watching. This motivates me to develop a ball tracking video. The pictures shown in this figure are captured with great details through tracking by detecting the ball. The right picture happened when game was over and players came up to surround the ball to celebrate. The ball was tracked until the tracker shifted to the whole group of people as the ball was invisible at that moment. But from the trajectory, the tracker knew it stayed there.

broadcasting and even satellites in the form of hyperspectral images). Some of these stream videos have become the resources for video streaming services such as online video sharing and live streams.

Sports videos. With so many potential applications in mind, the models developed in this dissertation are indeed motivated by such real-world scenarios. In this dissertation, neither I propose a model for the sake of proposing a new model nor I solve a problem without proposing any new model. The underlying research philosophy is that a model needs to be equipped together with a task potentially useful for a certain application.

Given those, I then think that it is worthy to spend efforts to design and refine

a model, develop algorithms for the model to work, analyze their convergence and optimality properties and even pursue theoretical guarantees. There are cases where real working algorithms could hardly be proved for and some other cases where algorithms with theoretical proofs do not work so well. I tend to more forgive the former cases but do respect the efforts spent on the latter cases by other researchers.

Although I do not aim at building a real working system myself, each piece of this dissertation work provides a prototype that is potentially useful for building real-world system. As another motivating example, sports videos have become the favorite of TV and Internet users who always expect even better visual experiences. For example, various techniques have been designed to improve the experience of watching soccer games through broadcast videos. Even for non-professional games as shown in Fig. 8.13.

As shown in Fig. 8.14, a moving camera prototype was developed to track a player lively through the video stream [272]. This prototype is not aimed to replace cameramen, who generally understand a game better than computers. However, there is a real-world problem - balls and players sometimes move so far away or move so fast that the camera is slower to catch up (say, during long shots, long passes or complex passes). The developed prototype is aimed to assist cameramen for such tasks which are easy for computers but challenging for cameramen. It also runs real-time credited to the parallel computing paradigm (see Fig. 8.15).

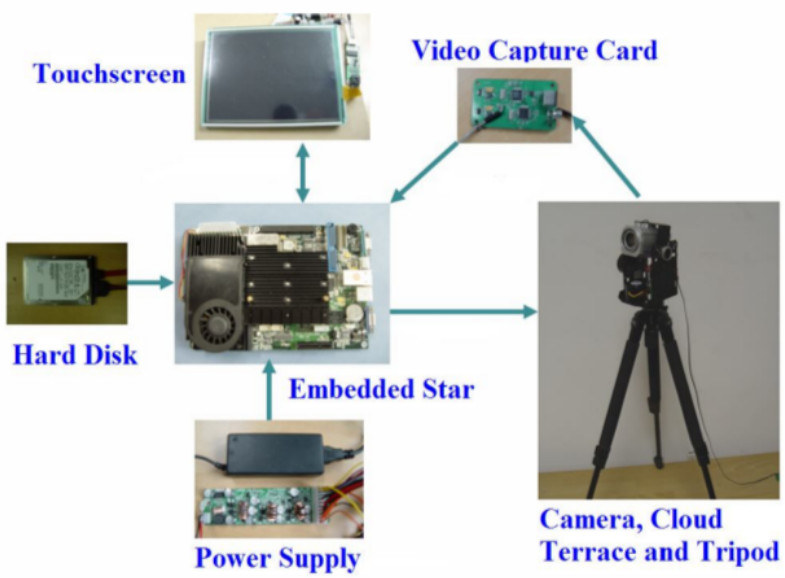


Figure 8.14: This target tracking camera is built using an embedded system platform named Embedded Star together with the camera, cloud terrace, video capture card, LCD touch screen, MCU, tripod, and other peripheral equipment.

Figure 8.15: Screen-shots of the person tracked video. It demonstrates that the tracking is robust even when the person is blocked by the tree.

Figure 8.16: Searching the goals in a game is still not available on ©YouTube. However, manual added time stamps of goals are available, which is hardly generalizable to large scale.

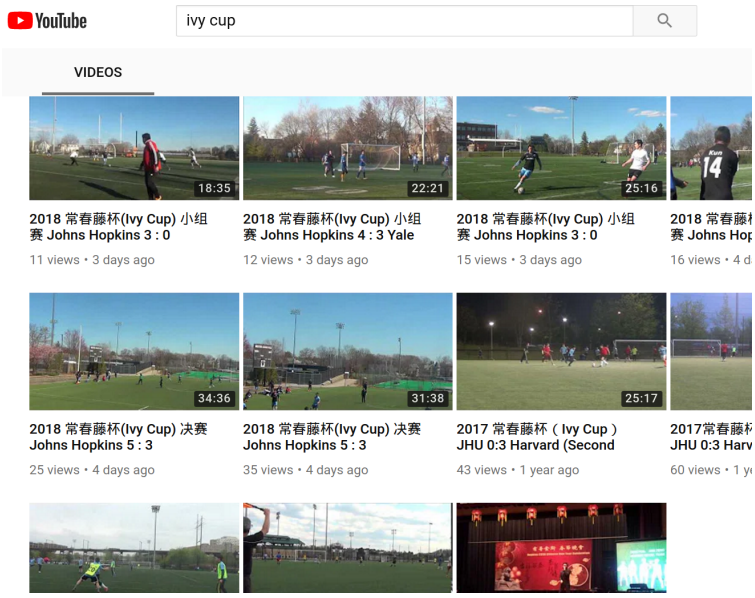


Figure 8.17: Searching videos using keyword is widely available. Results provided by ©YouTube are pretty accurate.

Video event retrieval. Particularly, finding a certain event in a video stream is highly demanded in surveillance and online video services. As shown in Fig. 8.16, video event retrieval is different from video retrieval, which is just searching a subset of videos out of a large set of videos as shown in Fig. 8.17.

Video analysis in the cloud. However, normally there is not too much video analysis computation happening on the camera platform except videos are restricted from transmitting due to privacy and network security issues.

Instead, videos are uploaded to video hosting servers or online video platforms on the fly. For example, the surveillance cameras pass the video stream to a server through WIFI. In the meanwhile, users can watch the video downloaded lively from the server. For the sports broadcasting videos, they are consolidated by professional editors lively and then media platforms such as TV channels, YouTube live and so on. The intermediate processing needs to happen in an efficient manner due to the users' expectation of low latency. Platforms such as Amazon Web Services (AWS, see Fig. 8.18) and Emotient Analytics (see Fig. 8.19) provides effective APIs to developers so that they can build video analysis algorithms without diving too much into the speeding up issue. These platforms are also great application scenarios where the models developed in this dissertation might be nicely applied to.

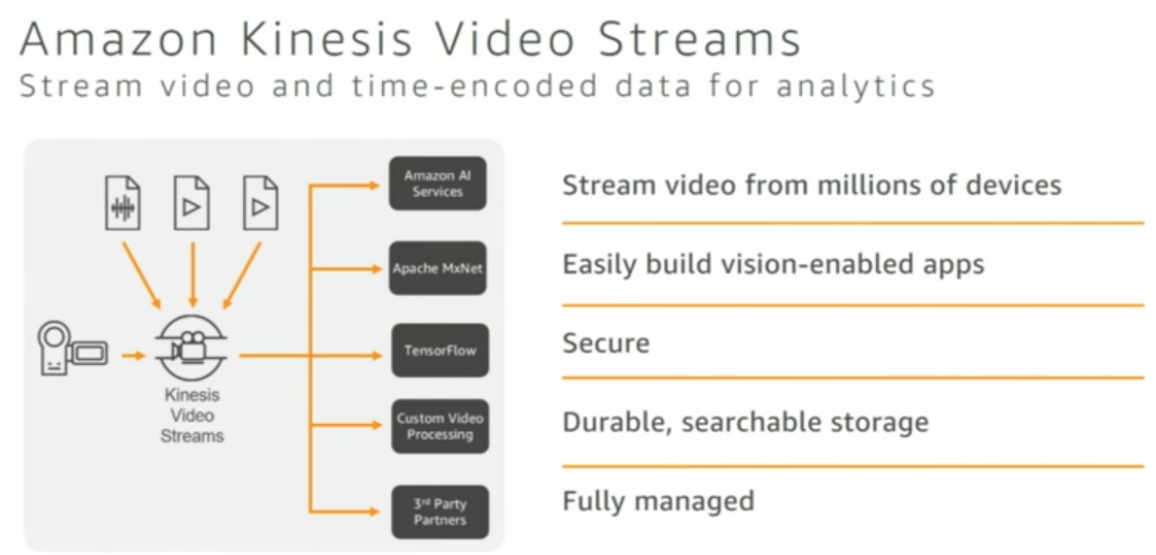


Figure 8.18: ©Amazon Kinesis Video Streams are a set of AWS APIs that support video streaming development.

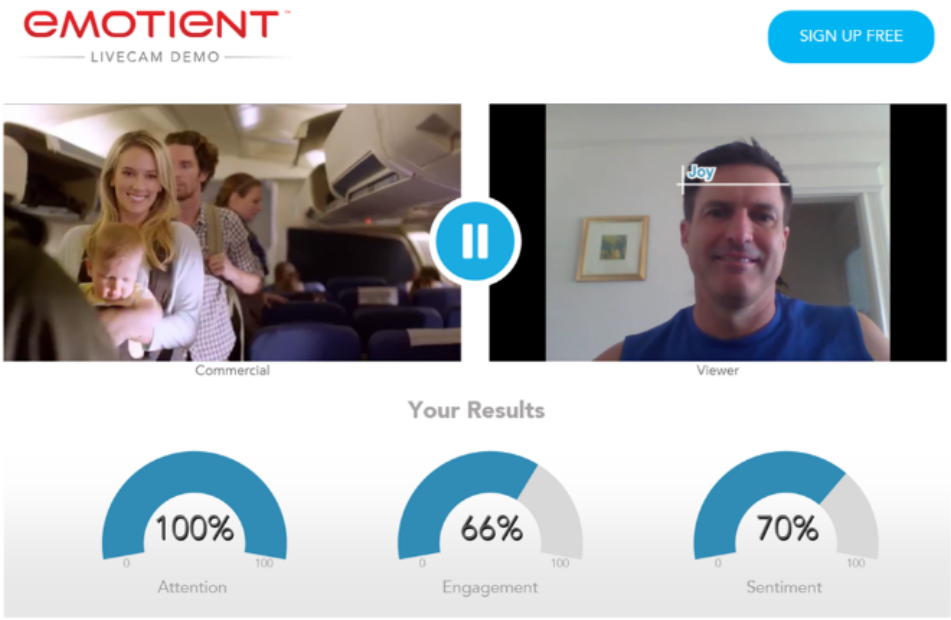


Figure 8.19: ©Emotient Analytics is a cloud platform for emotion and sentiment analysis. Developers upload videos and then use the provided APIs to process videos.

Bibliography

[1] J. C. Niebles, B. Han, A. Ferencz, and L. Fei-Fei, "Extracting moving people from Internet videos," in *ECCV*, 2008.

[2] B. Russell, A. Torralba, K. Murphy, and W. Freeman, "LabelMe: a database and Web-based tool for image annotation," *IJCV*, vol. 77, no. 1, pp. 157–173, 2008.

[3] G. Gkioxari and J. Malik, "Finding action tubes," in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 2015, pp. 759–768.

[4] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015.

[5] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard, "The megaface benchmark: 1 million faces for recognition at scale," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[6] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[7] Y. LeCun, C. Cortes, and C. Burges. (1998) The mnist database of handwritten digits. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[8] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool, "Creating summaries from user videos," in *European conference on computer vision*. Springer, 2014, pp. 505–520.

[9] M. J. Paul and M. Dredze, "You are what you tweet: Analyzing twitter for public health." 2011.

[10] X. Xiang, H. Chang, and J. Luo, "Online web-data-driven segmentation of selected moving objects in videos," in *Asian Conference on Computer Vision*. Springer, 2012, pp. 134–146.

[11] A. C. Sankaranarayanan, P. K. Turaga, R. Chellappa, and R. G. Baraniuk, "Compressive acquisition of linear dynamical systems," *SIAM Journal on Imaging Sciences*, vol. 6, no. 4, pp. 2109–2133, 2013.

[12] A. Ravichandran, R. Chaudhry, and R. Vidal, "Categorizing dynamic textures using a bag of dynamical systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 2, pp. 342–353, 2013.

[13] R. Hong, J. Tang, H.-K. Tan, S. Yan, C. Ngo, and T.-S. Chua, "Event driven summarization for web videos," in *Proceedings of the first SIGMM workshop on Social media*. ACM, 2009, pp. 43–48.

[14] C.-W. Ngo, Y.-F. Ma, and H.-J. Zhang, "Automatic video summarization by graph modeling," in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*. IEEE, 2003, pp. 104–109.

[15] Y.-F. Ma, L. Lu, H.-J. Zhang, and M. Li, "A user attention model for video summarization," in *Proceedings of the tenth ACM international conference on Multimedia*. ACM, 2002, pp. 533–542.

[16] T. Liu and J. R. Kender, "Optimization algorithms for the selection of key frame sequences of variable length," in *European Conference on Computer Vision*. Springer, 2002, pp. 403–417.

[17] Y. Pritch, A. Rav-Acha, A. Gutman, and S. Peleg, "Webcam synopsis: Peeking around the world," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE, 2007, pp. 1–8.

[18] H.-W. Kang and X.-Q. Chen, "Space-time video montage," in *computer vision and pattern recognition, 2006 IEEE computer society conference on*, vol. 2. IEEE, 2006, pp. 1331–1338.

[19] J. He, Z. Deng, M. Ibrahim, and G. Mori, "Generic tubelet proposals for action localization," in *WACV*. IEEE, 2018.

[20] Y. Ke, R. Sukthankar, and M. Hebert, "Event detection in crowded videos," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. IEEE, 2007, pp. 1–8.

[21] K. Soomro and M. Shah, "Unsupervised action discovery and localization in videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 696–705.

[22] L. Wang, G. Hua, R. Sukthankar, J. Xue, Z. Niu, and N. Zheng, "Video object discovery and co-segmentation with extremely weak supervision," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 10, pp. 2074–2088, 2017.

[23] X. Hong, H. Chang, S. Shan, X. Chen, and W. Gao, "Sigma set: A small second order statistical region descriptor," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 1802–1809.

[24] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

[25] M. Liu, S. Shan, R. Wang, and X. Chen, "Learning expressionlets on spatio-temporal manifold for dynamic facial expression recognition," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014, pp. 1749–1756.

[26] X. Hong, G. Zhao, M. Pietikäinen, and X. Chen, "Combining lbp difference and feature correlation for texture description," *IEEE Transactions on Image Processing*, vol. 23, no. 6, pp. 2557–2568, 2014.

[27] P. Zhu, W. Zuo, L. Zhang, S. C.-K. Shiu, and D. Zhang, "Image set-based collaborative representation for face recognition," *IEEE transactions on information forensics and security*, vol. 9, no. 7, pp. 1120–1132, 2014.

[28] M. Liu, R. Wang, S. Li, S. Shan, Z. Huang, and X. Chen, "Combining multiple kernel methods on riemannian manifold for emotion recognition in the wild," in *Proceedings of the 16th International Conference on Multimodal Interaction*. ACM, 2014, pp. 494–501.

[29] Z. Huang, R. Wang, S. Shan, and X. Chen, "Projection metric learning on grassmann manifold with application to video based face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 140–149.

[30] A. Rahimi, T. Darrell, and B. Recht, "Learning appearance manifolds from

video," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 868–875.

[31] Z. Huang, X. Zhao, S. Shan, R. Wang, and X. Chen, "Coupling alignments with recognition for still-to-video face recognition," in *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3296–3303.

[32] P. Zhu, L. Zhang, W. Zuo, and D. Zhang, "From point to set: Extend the learning of distance metrics," in *Computer Vision (ICCV), 2013 IEEE International Conference on*. IEEE, 2013, pp. 2664–2671.

[33] T.-K. Kim, J. Kittler, and R. Cipolla, "Discriminative learning and recognition of image set classes using canonical correlations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1005–1018, 2007.

[34] J. Hamm and D. D. Lee, "Grassmann discriminant analysis: a unifying view on subspace-based learning," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 376–383.

[35] O. Arandjelovic, G. Shakhnarovich, J. Fisher, R. Cipolla, and T. Darrell, "Face recognition with image sets using manifold density divergence," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 581–588.

[36] X. Xiang, "A brief review on visual tracking methods," in *Intelligent Visual*

Surveillance (IVS), 2011 Third Chinese Conference on. IEEE, 2011, pp. 41–44.

[37] R. Brunelli, *Template matching techniques in computer vision: theory and practice.* John Wiley & Sons, 2009.

[38] L. Matthews, T. Ishikawa, and S. Baker, "The template update problem," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 6, pp. 810–815, 2004.

[39] J. Shi *et al.*, "Good features to track," in *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on.* IEEE, 1994, pp. 593–600.

[40] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *International journal of computer vision*, vol. 56, no. 3, pp. 221–255, 2004.

[41] A. Doucet, S. Godsill, and C. Andrieu, "On sequential monte carlo sampling methods for bayesian filtering," *Statistics and computing*, vol. 10, no. 3, pp. 197–208, 2000.

[42] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[43] C. Lea, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks: A unified approach to action segmentation," in *proceedings of the Eurepean Conference on Computer Vision*, 2016, pp. 47–54.

[44] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[45] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[46] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in neural information processing systems*, 2014, pp. 568–576.

[47] H. Wang and L. Wang, "Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[48] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell, "ActionVLAD: Learning spatio-temporal aggregation for action classification," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[49] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD:

Cnn architecture for weakly supervised place recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5297–5307.

[50] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010, pp. 3304–3311.

[51] A. Diba, V. Sharma, and L. Van Gool, "Deep temporal linear encoding networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[52] W. Pei, T. Baltrusaitis, D. M. Tax, and L.-P. Morency, "Temporal attention-gated model for robust sequence classification," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[53] X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei, "Deep feature flow for video recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[54] R. Hou, C. Chen, and M. Shah, "Tube convolutional neural network (t-cnn) for action detection in videos," in *ICCV*. IEEE, 2017.

[55] K. Fang, Y. Xiang, and S. Savarese, "Recurrent autoregressive networks for online multi-object tracking," in *WACV*, 2017.

[56] Y.-W. Chao, Y. Liu, X. Liu, H. Zeng, and J. Deng, "Learning to detect human-object interactions," in *WACV*, 2017.

[57] W. T. T. D. H. G. D. Xiang, Xiang; Zhu, "Survey on multi-scale cnns for lung nodule detection," in *IEEE International Symposium on Biomedical Imaging*, 2018.

[58] X. T. T. D. H. G. D. X. X. Zhu, Wentao; Xiang, "Adversarial deep structured nets for mass segmentation from mammograms," in *Proceedings of the IEEE International Symposium on Biomedical Imaging*, 2018.

[59] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[60] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2625–2634.

[61] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional network," in *proceedings of the IEEE International Conference on Computer Vision*, 2015, p. 44894497.

[62] P. Parmar and B. Morris, "Learning to score olympic events," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017.

[63] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4724–4733.

[64] G. A. Sigurdsson, S. Divvala, A. Farhadi, and A. Gupta, "Asynchronous temporal fields for action recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[65] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Temporal residual networks for dynamic scene recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[66] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-rnn: Deep learning on spatio-temporal graphs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5308–5317.

[67] G. Garcia-Hernando and T.-K. Kim, "Transition forests: Learning discriminative temporal transitions for action recognition and detection," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[68] A. Miech, I. Laptev, and J. Sivic, "Learnable pooling with context gating for video classification," *arXiv preprint arXiv:1706.06905*, 2017.

[69] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE Trans. Inf. Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[70] E. J. Candes, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Comm. Pure Appl. Math.*, vol. 59, pp. 1207–1223, 2006.

[71] ——, "Robust uncertainty principles: Exact signal reconstruction from highly incomplet frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[72] K. Huang and S. Aviyente, "Sparse representations for signal classification," in *NIPS*, 2006.

[73] J. Wright, A. Yang, A. Ganesh, S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE T-PAMI*, vol. 31, no. 2, pp. 210–227, 2009.

[74] J. Friedman, T. Hastie, and R. Tibshirani, "A Note on the Group Lasso and a Sparse Group Lasso," in *arXiv*, vol. 1001.0736, 2010.

[75] Y. C. Eldar and H. Rauhut, "Average case analysis of multichannel sparse recovery using convex relaxation," *IEEE Trans. Inf. Theory*, vol. 56, no. 1, pp. 505–519, 2010.

[76] J. Huang and T. Zhang, "The benefit of group sparsity," *The Annals of Statistics*, vol. 38, no. 4, pp. 1978–2004, 2010.

[77] E. Candes, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3.

[78] P. Sprechmann, I. Ramrez, G. Sapiro, and Y. Eldar, "C-HiLasso: A collaborative hierarchical sparse modeling framework," *IEEE T-SP*, vol. 59, no. 9, pp. 4183–4198, 2011.

[79] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE T-PAMI*, vol. 35, no. 11, pp. 2765–2781, 2013.

[80] Wikipedia, "Tikhonov regularization," http://en.wikipedia.org/wiki/Tikhonov_regularization.

[81] Z. Wang, J. Yang, N. Nasrabadi, and T. Huang, "A max-margin perspective on sparse representation-based classification," in *IEEE ICCV*, 2013.

[82] Y. Suo, M. Dao, U. Srinivas, V. Monga, and T. D. Tran, "Structured dictionary learning for classification," in *arXiv*, vol. 1406.1943, 2014.

[83] L. Zhang, M. Yang, and X. Feng, "Sparse representation or collaborative representation: Which helps face recognition?" in *IEEE ICCV*, 2011.

[84] J. Wright, A. Ganesh, A. Yang, Z. Zhou, and Y. Ma, "A tutorial on how to apply the models and tools correctly," in *arXiv*, vol. 1111.1014, 2011.

[85] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via Orthogonal Matching Pursuit," *IEEE Trans. Inf. Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[86] E. Amaldi and V. Kann, "On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems," *Theoretical Computer Science*, vol. 209, pp. 237–260, 1998.

[87] A. Y. Yang, A. Ganesh, Z. Zhou, S. Sastry, and Y. Ma, "Fast l1-minimization algorithms for robust face recognition," *IEEE Trans. Image Proc.*, vol. 22, no. 8, pp. 3234–3245, 2013.

[88] G. Liu, Z. Liu, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE T-PAMI*, vol. 35, no. 1, pp. 171–185, 2013.

[89] S. Chen and D. Donoho, "Basis pursuit," in *Signals, Systems and Computers, 1994. 1994 Conference Record of the Twenty-Eighth Asilomar Conference on*, vol. 1. IEEE, pp. 41–44.

[90] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado, "Sparse solutions to linear inverse problems with multiple measurement vectors," *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2477–2488, 2005.

[91] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *CVPR*, 1991, pp. 586–591.

[92] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[93] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng, "Parsing natural scenes and natural language with recursive neural networks," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 129–136.

[94] Z. Huang, R. Wang, S. Shan, and X. Chen, "Face recognition on large-scale video in the wild with hybrid euclidean-and-riemannian metric learning," *Pattern Recognition*, vol. 48, no. 10, pp. 3113–3124, 2015.

[95] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[96] E. Learned-Miller, G. B. Huang, A. RoyChowdhury, H. Li, and G. Hua, "Labeled faces in the wild: A survey," *Advances in Face Detection and Facial Image Analysis*, pp. 189–248, 2016.

[97] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE International Conference on Computer Vision*, 2014.

[98] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in Neural Information Processing Systems*, 2014.

[99] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *British Machine Vision Conference*, 2015.

[100] X. Xiang, M. Dao, G. D. Hager, and T. D. Tran, "Hierarchical sparse and collaborative low-rank representation for emotion recognition," in *ICASSP*, 2015, pp. 3811–3815.

[101] Z. Huang, S. Shan, R. Wang, H. Zhang, S. Lao, A. Kuerban, and X. Chen, "A benchmark and comparative study of video-based face recogni-tion on cox face database," *IEEE Transaction on Image Processing*, vol. 24, pp. 5967–5981, 2015.

[102] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv:1408.5093*, 2014.

[103] H. Ding, S. K. Zhou, and R. Chellappa, "Facenet2expnet: Regularizing a deep face recognition net for expression recognition," in *Proceedings of the IEEE International Conference on Automatic Face & Gesture Recognition*, 2017.

[104] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: L 2 hypersphere

embedding for face verification," in *Proceedings of the 2017 ACM on Multimedia Conference*. ACM, 2017, pp. 1041–1049.

[105] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *Proceedings of the European Conference on Computer Vision*, 2016, pp. 499–515.

[106] R. Zhao, Q. Gan, S. Wang, and Q. Ji, "Facial expression intensity estimation using ordinal information," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3466–3474.

[107] J. Zhou, X. Hong, F. Su, and G. Zhao, "Recurrent convolutional neural network regression for continuous pain intensity estimation in video," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 84–92.

[108] C. Fabian Benitez-Quiroz, R. Srinivasan, and A. M. Martinez, "Emotionet: An accurate, real-time algorithm for the automatic annotation of a million facial expressions in the wild," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5562–5570.

[109] A. Dapogny, K. Bailly, and S. Dubuisson, "Pairwise conditional random forests for facial expression recognition," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3783–3791.

[110] Z. Wang, S. Wang, and Q. Ji, "Capturing complex spatio-temporal relations among facial muscles for facial expression recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 3422–3429.

[111] Y. Guo, G. Zhao, and M. Pietikäinen, "Dynamic facial expression recognition using longitudinal facial expression atlases," in *Proceedings of the European Conference on Computer Vision*, 2012, pp. 631–644.

[112] O. Rudovic, V. Pavlovic, and M. Pantic, "Multi-output laplacian dynamic ordinal regression for facial expression recognition and intensity estimation," in *CVPR*, 2012, pp. 2634–2641.

[113] M. Kim and V. Pavlovic, "Structured output ordinal regression for dynamic facial emotion intensity prediction," in *ECCV*, 2010, pp. 649–662.

[114] P. Yang, Q. Liu, and D. N. Metaxas, "Exploring facial expressions with compositional features," in *CVPR*, 2010, pp. 2638–2644.

[115] C. Fabian Benitez-Quiroz, R. Srinivasan, and A. M. Martinez, "Emotionet: An accurate, real-time algorithm for the automatic annotation of a million facial expressions in the wild," in *CVPR*, 2016.

[116] S. Rifai, Y. Bengio, A. Courville, P. Vincent, and M. Mirza, "Disentangling

factors of variation for facial expression recognition," in *ECCV*, 2012, pp. 808–822.

[117] X. Zhao, X. Liang, L. Liu, T. Li, Y. Han, N. Vasconcelos, and S. Yan, "Peak-piloted deep network for facial expression recognition," in *ECCV*, 2016, pp. 425–442.

[118] H. Jung, S. Lee, J. Yim, S. Park, and J. Kim, "Joint fine-tuning in deep neural networks for facial expression recognition," in *ICCV*, 2015.

[119] P. Liu, S. Han, Z. Meng, and Y. Tong, "Facial expression recognition via a boosted deep belief network," in *CVPR*, 2014, pp. 1805–1812.

[120] H. Chen, J. Li, F. Zhang, Y. Li, and H. Wang, "3d model-based continuous emotion recognition," in *CVPR*, 2015.

[121] Z. Zhang, J. M. Girard, Y. Wu, X. Zhang, P. Liu, U. Ciftci, S. Canavan, M. Reale, A. Horowitz, H. Yang, J. F. Cohn, Q. Ji, and L. Yin, "Multimodal spontaneous emotion corpus for human behavior analysis," in *CVPR*, 2016.

[122] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[123] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[124] T. Salimans and D. P. Kingma, "Weight normalization: A simple reparameterization to accelerate training of deep neural networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 901–901.

[125] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[126] Y. Liu, H. Li, and X. Wang, "Learning deep features via congenerous cosine loss for person recognition," *arXiv preprint arXiv:1702.06890*, 2017.

[127] R. Ranjan, C. D. Castillo, and R. Chellappa, "L2-constrained softmax loss for discriminative face verification," *arXiv preprint arXiv:1703.09507*, 2017.

[128] Z. Y. M. L. B. R. Weiyang Liu, Yandong Wen and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017.

[129] W. Liu, Y. Wen, Z. Yu, and M. Yang, "Large-margin softmax loss for convolutional neural networks," in *International Conference on Machine Learning*, 2016, pp. 507–516.

[130] J. M. S. G. L. C. Md. Abul Hasnat, Julien Bohn, "von mises-fisher mixture model-based deep learning: Application to face verification," *arXiv preprint arXiv:1706.04264*, 2017.

[131] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning

applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[132] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in neural information processing systems*, 2014, pp. 1988–1996.

[133] X. Wu, R. He, and Z. Sun, "A lightened cnn for deep face representation," *arXiv preprint arXiv:1511.02683*, 2015.

[134] W. Rudin *et al.*, *Principles of mathematical analysis, Chapter 10.* McGraw-Hill New York, 1964, vol. 3.

[135] R. Girshick, "Fast R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[136] P. Lucey, J. F. Cohn, K. M. Prkachin, P. E. Solomon, and I. Matthews, "Painful data: The unbc-mcmaster shoulder pain expression archive database," in *Proceedings of the IEEE International Conference on Automatic Face & Gesture Recognition*, 2011, pp. 57–64.

[137] O. Rudovic, V. Pavlovic, and M. Pantic, "Automatic pain intensity estimation with heteroscedastic conditional ordinal random fields," in *Proceedings of the International Symposium on Visual Computing*, 2013, pp. 234–243.

[138] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," *arXiv preprint:1411.7923*, 2014.

[139] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, 2016.

[140] X. Xiang and T. D. Tran, "Pose-selective max pooling for measuring similarity," in *ICPR workshops*, 2016.

[141] Y. Zhang and A. M. Martínez, "A weighted probabilistic approach to face recognition from multiple images and video sequences," *Image and Vision Computing*, vol. 24, no. 6, pp. 626–638, 2006.

[142] J. Stallkamp, H. K. Ekenel, and R. Stiefelhagen, "Video-based face recognition on real-world data," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on.* IEEE, 2007, pp. 1–8.

[143] A. Mian, "Unsupervised learning from local features for video-based face recognition," in *Automatic Face & Gesture Recognition, 2008. FG'08. 8th IEEE International Conference on.* IEEE, 2008, pp. 1–6.

[144] A. Hadid and M. Pietikainen, "Selecting models from videos for appearance-based face recognition," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 1. IEEE, 2004, pp. 304–308.

[145] W. Fan and D.-Y. Yeung, "Face recognition with image sets using hierarchically extracted exemplars from appearance manifolds," in *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on.* IEEE, 2006, pp. 177–182.

[146] D. Thomas, K. W. Bowyer, and P. J. Flynn, "Multi-frame approaches to improve face recognition," in *Motion and Video Computing, 2007. WMVC'07. IEEE Workshop on.* IEEE, 2007, pp. 19–19.

[147] U. Park, A. K. Jain, and A. Ross, "Face recognition in video: Adaptive fusion of multiple matchers," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on.* IEEE, 2007, pp. 1–8.

[148] L. Pigou, A. van den Oord, S. Dieleman, M. V. Herreweghe, and J. Dambre, "Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video," *arxiv*, June 2015.

[149] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in visual recognition," in *Proceedings of the International Conference on Machine Learning*, 2010.

[150] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[151] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "Youtube-8m: A large-scale video classification benchmark," *arxiv: 1609.08675*, September 2016.

[152] N. Crosswhite, J. Byrne, O. M. Parkhi, C. Stauffer, Q. Cao, and A. Zisserman, "Template adaptation for face verification and identification," *arxiv*, April 2016.

[153] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2625–2634.

[154] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. Mooney, and K. Saenko, "Translating videos to natural language using deep recurrent neural networks," in *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2014.

[155] F. Wang, X. Xiang, C. Liu, T. D. Tran, A. Reiter, G. D. Hager, H. Quon, J. Cheng, and A. L. Yuille, "Regularizing face verification nets for pain intensity regression," in *Image Processing (ICIP), 2017 IEEE International Conference on.* IEEE, 2017, pp. 1087–1091.

[156] S. Taheri, V. M. Patel, and R. Chellappa, "Component-based recognition of

faces and facial expressions," *IEEE Trans. on Affective Computing*, vol. 4, no. 4, pp. 360–371, 2013.

[157] S. Zafeiriou and M. Petrou, "Sparse representations for facial expressions recognition via l1 optimization," in *CVPR Workshop*, 2010.

[158] P. Lucey, J. Cohn, T. Kanade, J. Saragih, and Z. Ambadar, "The Extended CK Dataset (CK+): A complete dataset for action unit and emotion-specified expression," in *CVPR*, 2010.

[159] S. Chew, P. Lucey, S. Lucey, J. Saragih, J. Cohn, and S. Sridharan, "Person-independent facial expression detection using constrained local models," in *FG*, 2011, pp. 915–920.

[160] R. Walecki, O. Rudovic, V. Pavlovic, and M. Pantic, "Variable-state latent conditional random field models for facial expression analysis," *Image and Vision Computing*, 2016.

[161] S. Jain, C. Hu, and J. K. Aggarwal, "Facial expression recognition with temporal modeling of shapes," in *ICCV Workshops*, 2011, pp. 1642–1649.

[162] M. Liu, S. Li, S. Shan, R. Wang, and X. Chen, "Deeply learning deformable facial action parts model for dynamic expression analysis," in *ACCV*, 2014, pp. 143–157.

[163] A. C. Sankaranarayanan, P. K. Turaga, R. G. Baraniuk, and R. Chellappa,

"Compressive acquisition of dynamic scenes," in *European Conference on Computer Vision*. Springer, 2010, pp. 129–142.

[164] P. Liu, J. T. Zhou, I. W.-H. Tsang, Z. Meng, S. Han, and Y. Tong, "Feature disentangling machine-a novel approach of feature selection and disentangling in facial expression analysis," in *ECCV*, 2014, pp. 151–166.

[165] S. Reed, K. Sohn, Y. Zhang, and H. Lee, "Learning to disentangle factors of variation with manifold interaction," in *ICML*, 2014, pp. 1431–1439.

[166] G. D. Hager and P. N. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE T-PAMI*, vol. 20, no. 10, pp. 1025–1039, 1998.

[167] X. Xiang and T. D. Tran, "Linear disentangled representation learning for facial actions," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[168] ——, "Recursively measured action units," in *ICPR Workshops*, 2016.

[169] K. Mori, D. Deguchi, K. Akiyama, T. Kitasaka, C. R. Maurer, Y. Suenaga, H. Takabatake, M. Mori, and H. Natori, "Hybrid bronchoscope tracking using a magnetic tracking sensor and image rgistration," in *MICCAI*, 2005.

[170] D. Mirota, H. Wang, R. H. Taylor, M. Ishii, G. L. Gallia, and G. D. Hager, "A

system for video-based navigation for endoscopic endonasal skull base surgery," *IEEE Trans. Med. Imaging*, vol. 31, pp. 963–976, 2012.

[171] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, *An Invitation to 3-D Vision*. Springer, 2004.

[172] C. Wu, "VisualSFM: A Visual Structure from Motion System," http://ccwu.me/vsfm/, 2011.

[173] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, ISBN: 0521540518, 2004.

[174] H. N. Tokgozoglu, E. M. Meisner, M. Kazhdan, and G. D. Hager, "Color-based hybrid reconstruction for endoscopy," in *CVPR Workshops*, 2012.

[175] T. Collins and A. Bartoli, "Towards live monocular 3d laparoscopy using shading and specularity information," in *IPCAI*, 2012.

[176] D. Nister, "An efficient solution to the five-point relative pose problem," *IEEE T-PAMI*, vol. 26.

[177] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," in *BMVC*, 2002.

[178] D. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, 2005.

[179] G. A. Puerto-Souza and G. L. Mariottini, "Adaptive multi-affine (ama) feature-matching algorithm and its application to minimally-invasive surgery images," in *MICCAI*, 2012.

[180] ——, "Hierarchical Multi-Affine (HMA) algorithm for fast and accurate feature matching in Minimally-Invasive surgical images," in *IEEE IROS*, 2012.

[181] G. A. Puerto and G. L. Mariottini, "A fast and accurate feature-matching algorithm for minimally invasive endoscopic images," *IEEE Transactions on Medical Imaging*, 2013.

[182] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2010.

[183] D. Abretske, D. Mirota, G. D. Hager, and M. Ishii, "Intelligent frame selection for anatomic reconstruction from endoscopic video," in *WACV 2009*, 2009.

[184] D. Mirota, *Video-Based Navigation with Application to Endoscopic Skull Base Surgery*. Johns Hopkins University Computer Science Department Ph.D. Dissertation, 2012.

[185] Wikipedia, "Quaternion," http://en.wikipedia.org/wiki/Quaternion.

[186] ——, "Euler Angles," http://en.wikipedia.org/wiki/Euler_angles.

[187] ——, "Rotation Formalisms in Three Dimensions," http://en.wikipedia.org/wiki/Rotation_formalisms_in_three_dimensions.

[188] ——, "Euler's Rotation Theorem," http://en.wikipedia.org/wiki/Euler's_rotation_theorem.

[189] Caltech Vision Lab, "Camera Calibration Toolbox for Matlab," http://www.vision.caltech.edu/bouguetj/calib_doc/, July 2010.

[190] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," http://www.vlfeat.org/, 2008.

[191] G. A. Puerto and G. L. Mariottini, "HMA Feature-Matching Toolbox," http://ranger.uta.edu/~gianluca/feature_matching/, 2012.

[192] P. Torr, "Structure and motion toolkit," http://www.mathworks.com, 2004.

[193] H. Grabner and H. Bischof, "On-line boosting and vision," in *IEEE CVPR*, 2006.

[194] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online Multiple Instance Learning," *IEEE T-PAMI*, vol. 33, no. 8, pp. 1619–1632, 2011.

[195] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-N learning: bootstrapping binary classifiers by structural constraints," in *IEEE CVPR*, 2010.

[196] M. Yang, J. Yuan, and Y. Wu, "Spatial selection for attentional visual tracking," in *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on.* IEEE, 2007, pp. 1–8.

[197] J. Fan, Y. Wu, and S. Dai, "Discriminative spatial attention for robust tracking," in *European Conference on Computer Vision*. Springer, 2010, pp. 480–493.

[198] K. Fragkiadaki and J. Shi, "Detection free tracking: Exploiting motion and topology for segmenting and tracking under entanglement," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 2073–2080.

[199] N. C. Oza, "Online ensemble learning," in *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*. AAAI Press, 2000, p. 1109.

[200] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial intelligence*, vol. 89, no. 1-2, pp. 31–71, 1997.

[201] C. Zhang, J. C. Platt, and P. A. Viola, "Multiple instance boosting for object detection," in *Advances in neural information processing systems*, 2006, pp. 1417–1424.

[202] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[203] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *European conference on computer vision*. Springer, 2008, pp. 234–247.

[204] R. Hess and A. Fern, "Discriminatively trained particle filters for complex multi-object tracking," in *IEEE CVPR*, 2009.

[205] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *IEEE CVPR*, 2006.

[206] X. Xiang, D. Mirota, A. Reiter, and G. D. Hager, "Is multi-model feature matching better for endoscopic motion estimation?" in *MICCAI workshops*, 2014.

[207] J. Malcolm, Y. Rathi, and A. Tannenbaum, "Tracking through clutter using graph cuts," in *BMVC*, 2007.

[208] B. L. Price, B. S. Morse, and S. Cohen, "Livecut: Learning-based interactive video segmentation by evaluation of multiple propagated cues," in *IEEE ICCV*, 2009.

[209] A. Fathi, M. Balcan, X. Ren, and J. Rehg, "Combining self training and active learning for video segmentation," in *BMVC*, 2011.

[210] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman, "Segmenting scenes by matching image composites," in *NIPS*, 2009.

[211] N. Ikizler-Cinbis, R. G. Cinbis, and S. Sclaroff, "Learning Actions From the Web," in *IEEE ICCV*, 2009.

[212] M. Rodriguez, J. Sivic, I. Laptev, and J.-Y. Audibert, "Data-driven crowd analysis in videos," in *IEEE ICCV*, 2011.

[213] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari, "Learning object class detectors from weakly annotated video," in *IEEE CVPR*, 2012.

[214] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov, "Bilayer segmentation of live video," in *IEEE CVPR*, 2006.

[215] Y. Li, J. Sun, and H.-Y. Shum, "Video object cut and paste," *ACM Transaction on Graphics*, vol. 24, no. 3, pp. 595–600, 2005.

[216] J. Wang, P. Bhat, A. Colburn, M. Agrawala, and M. Cohen, "Interactive video cutout," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 585–594, 2005.

[217] Y. Boykov and M. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images," in *IEEE ICCV*, 2001.

[218] X. Ren and J. Malik, "Tracking as repeated figure/ground segmentation," in *IEEE CVPR*, 2007.

[219] Z. Yin and R. T. Collins, "Online figure-ground segmentation with edge pixel classification," in *BMVC*, 2008.

[220] A. Bugeau and P. Perez, "Detection and segmentation of moving objects in highly dynamic scenes," in *IEEE CVPR*, 2007.

[221] J. C. Niebles, B. Han, and L. Fei-Fei, "Efficient extraction of human motion volumes by tracking," in *IEEE CVPR*, 2010.

[222] J. Malcolm, Y. Rathi, and A. Tannenbaum, "Graph cut segmentation with nonlinear shape priors," in *IEEE ICIP*, 2007.

[223] Z. Yin and R. T. Collins, "Shape constrained figure-ground segmentation and tracking," in *IEEE CVPR*, 2009.

[224] N. Vu and B. Manjunath, "Shape prior segmentation of multiple objects with graph cuts," in *IEEE CVPR*, 2008.

[225] J. Cui, Q. Yang, F. Wen, Q. Wu, C. Zhang, L. V. Gool, and X. Tang, "Transductive object cutout," in *IEEE CVPR*, 2008.

[226] F. Zhong, X. Qin, and Q. Peng, "Transductive segmentation of live video with non-stationary background," in *IEEE CVPR*, 2010.

[227] V. Vineet, J. Warrell, L. Ladicky, and P. Torr., "Human instance segmentation from video using detector-based crf," in *BMVC*, 2011.

[228] Y. J. Lee and K. Grauman, "Collect-cut: Segmentation with top-down cues discovered in multi-object images," in *IEEE CVPR*, 2010.

[229] S. Dambreville, Y. Rathi, and A. Tannenbaum, "A framework for image segmentation using shape models and kernel space shape priors," *T-PAMI*, vol. 30, no. 8, pp. 1385–1399, 2008.

[230] Y. J. Lee, J. Kim, and K. Grauman, "Key-segments for video object segmentation," in *IEEE CVPR*, 2012.

[231] S. Mika, B. Scholkopf, A. Smola, K.-R. Muller, M. Scholz, and G. Ratsch, "Kernel PCA and de-noising in feature spaces," in *NIPS*, 1999.

[232] C. Li, C. Xu, C. Gui, and M. Fox, "Level set evolution without re-initialization: A new variational formulation," in *IEEE CVPR*, 2005.

[233] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE T-PAMI*, vol. 26, no. 9, pp. 1124–1137, 2004.

[234] S. S. Vedula, A. Malpani, N. Ahmidi, S. Khudanpur, G. Hager, and C. C. G. Chen, "Task-level vs. segment-level quantitative metrics for surgical skill assessment," *Journal of Surgical Education*, vol. 73, no. 3, pp. 482–489, 2016.

[235] K. Wnuk and S. Soatto, "Analyzing diving: a dataset for judging action quality," in *Asian Conference on Computer Vision*. Springer, 2010, pp. 266–276.

[236] H. Pirsiavash, C. Vondrick, and A. Torralba, "Assessing the quality of actions,"

in *proceedings of the Eurepean Conference on Computer Vision*, 2014, pp. 556–571.

[237] V. Venkataraman, I. Vlachos, and P. Turaga, "Dynamical regularity for action analysis," in *proceedings of the British Machine Vision Conference*, 2015.

[238] P. Parmar and B. T. Morris, "Measuring the quality of exercises," in *Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the*. IEEE, 2016, pp. 2241–2244.

[239] A. Zia, Y. Sharma, V. Bettadapura, E. L. Sarin, M. A. Clements, and I. Essa, "Automated assessment of surgical skills using frequency analysis," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 430–438.

[240] F. Wang, X. Xiang, C. Liu, T. D. Tran, A. Reiter, G. D. Hager, H. Quon, J. Cheng, and A. L. Yuille, "Regularizing face verification nets for pain intensity regression," in *proceedings of the IEEE Conference on Image Processing*, 2017.

[241] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3d residual networks," in *proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5534–5542.

[242] L. Sun, K. Jia, D.-Y. Yeung, and B. E. Shi, "Human action recognition using

factorized spatio-temporal convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4597–4605.

[243] B. D. Haeffele and R. Vidal, "Global optimality in neural network training," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[244] B. Haeffele, E. Young, and R. Vidal, "Structured low-rank matrix factorization: Optimality, algorithm, and applications to image processing," in *International Conference on Machine Learning*, 2014, pp. 2007–2015.

[245] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional lstm networks for improved phoneme classification and recognition," *proceedings of the International Conference on Artificial Neural Networks*, 2005.

[246] L. Ding and C. Xu, "Tricornet: A hybrid temporal convolutional and recurrent network for video action segmentation," *arXiv preprint:1705.07818*, 2017.

[247] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng, "Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 3361–3368.

[248] X. Xiang, Y. Tian, A. Reiter, G. D. Hager, and T. D. Tran, "S3d: Stacking segment-level p3d for action quality assessment," in *IEEE ICIP*, 2018.

[249] G. Lin, C. Shen, D. Suter, and A. Hengel, "A general two-step approach to learning-based hashing," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2552–2559.

[250] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," in *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on.* IEEE, 2006, pp. 459–468.

[251] A. Gionis, P. Indyk, R. Motwani *et al.*, "Similarity search in high dimensions via hashing," in *VLDB*, vol. 99, no. 6, 1999, pp. 518–529.

[252] M. Raginsky and S. Lazebnik, "Locality-sensitive binary codes from shift-invariant kernels," in *Advances in neural information processing systems*, 2009, pp. 1509–1517.

[253] R. Salakhutdinov and G. Hinton, "Semantic hashing," *International Journal of Approximate Reasoning*, vol. 50, no. 7, pp. 969–978, 2009.

[254] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Advances in neural information processing systems*, 2008, pp. 1753–1760.

[255] Y. Gong and S. Lazebnik, "Iterative quantization: A procrustean approach to learning binary codes," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on.* IEEE, 2011, pp. 817–824.

[256] W. Kong and W.-J. Li, "Isotropic hashing," in *Advances in Neural Information Processing Systems*, 2012, pp. 1646–1654.

[257] F. Shen, C. Shen, W. Liu, and H. Tao Shen, "Supervised discrete hashing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 37–45.

[258] J. Wang, S. Kumar, and S.-F. Chang, "Sequential projection learning for hashing with compact codes," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 1127–1134.

[259] M. Norouzi and D. M. Blei, "Minimal loss hashing for compact binary codes," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 353–360.

[260] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2074–2081.

[261] P. Zhang, W. Zhang, W.-J. Li, and M. Guo, "Supervised hashing with latent factor models," in *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. ACM, 2014, pp. 173–182.

[262] W.-C. Kang, W.-J. Li, and Z.-H. Zhou, "Column sampling based discrete supervised hashing," in *AAAI*, 2016.

[263] G. Lin, C. Shen, Q. Shi, A. Hengel, and D. Suter, "Fast supervised hashing with decision trees for high-dimensional data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1963–1970.

[264] Y. Weiss, R. Fergus, and A. Torralba, "Multidimensional spectral hashing," in *European Conference on Computer Vision*. Springer, 2012, pp. 340–353.

[265] Y. Gong, S. Lazebnik, A. Gordo, and F. Perronnin, "Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 12, pp. 2916–2929, 2013.

[266] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*. Springer, 2010, pp. 177–186.

[267] K. Lange, D. R. Hunter, and I. Yang, "Optimization transfer using surrogate objective functions," *Journal of computational and graphical statistics*, vol. 9, no. 1, pp. 1–20, 2000.

[268] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[269] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y. Zheng, "Nus-wide: a real-world web image database from national university of singapore," in *Proceedings*

of the ACM international conference on image and video retrieval. ACM, 2009, p. 48.

[270] H. Zhu, F. Wang, X. Xiang, and T. D. Tran, "Supervised hashing with jointly learning embedding and quantization," in *Image Processing (ICIP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 3715–3719.

[271] M. Xu, A. Sharghi, X. Chen, and D. J. Crandall, "Fully-coupled two-stream spatiotemporal networks for extremely low resolution action recognition," in *IEEE WACV*, 2018.

[272] X. Xiang, W. Chen, and D. Zeng, "Intelligent target tracking and shooting system with mean shift," in *Parallel and Distributed Processing with Applications, 2008. ISPA'08. International Symposium on*. IEEE, 2008, pp. 417–421.

Vita

Xiang Xiang is a PhD student in Computer Science at Johns Hopkins University, Baltimore, USA, since 2012 with Gregory D. Hager (initial appointed advisor since 2012) and a PhD candidate in Computer Science since 2014 with Trac D. Tran (primary advisor since 2014) and Gregory D. Hager (co-advisor since 2014) co-listed as my official advisors. He received the B.S. degree from Wuhan University, Wuhan, China, in 2009, the M.S. degree from Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2012, and the M.S.E. degree from Johns Hopkins University in 2014, all in Computer Science. His research interests are computer vision and machine learning with a focus on representation learning for video understanding, facial analysis, affective computing and bio-medical applications.

Since coming to US, I have fortunately experienced various kinds of working places including a successful startup, a big company's industrial lab and a federal government research lab. In the past, I worked at the Imaging Division of FDA's Center for Device and Radiology, Silver Spring, MD during June - Aug., 2017 as an Oak Ridge Institute's fellow , at Phillips Research North America, Cambridge, MA during May

VITA

- Aug., 2016 as a research intern on realtime patient recognition for Philips's Vital Signs Camera, at Emotient RD (now Apple AI), San Diego, CA during June-Aug. 2015 as a research intern on advertisement video analysis with Ian Fasel and Javier Movellan. Right before coming to Hopkins, I graduated in 2012 from the Institute of Computing Technology at Chinese Academy of Sciences, advised by Xilin Chen and also working with Hong Chang and Shiguang Shan at the Visual Information Processing and Learning (VIPL) group.

My dissertation research focuses on the intersection between computer vision and matrix theory. In particular, solving visual recognition using representation learning involves discovering the data structure. A data structure, in the simplest terms, is a structured representation of information. For example, video frames are highly correlated so that an underlying low-rank matrix exist if we represent each frame as a vector and arrange them to form a matrix. Low-rank matrix approximation does not necessary give us a visually-meaningful object so once again we apply other prio r knowledge such as structured sparsity in the representation learning. The representation of objects such as the face and human body helps computer programs percept them in the real world. Observing that the same object (input space) has various attributes (output space), I've been interested in robust mappings with attribute disentanglement in real-world visual data such as videos of faces. Identity-labeled data are rich while quantified expression labels are not. Can we transfer a well-trained face recognizer to expression? Yes! In the linear function space, one observation is

that a specific expressive face can be a linear combination of various people's faces with the same expression. In the non-linear space, I show how to design a deep network to learn an expression recognizer by refining a deep face recognizer with a few additional training examples with expression labels. These research findings are generalizable to other couples of attributes such as identity and pose. Similarly, a specific person's face at a specific pose can be represented using a bag of the same person's faces at various poses. Beyond discriminative models, the learned knowledge relating attributes can serve as the prior knowledge to "drive" the object to "move" from a static image in novel ways, namely to generate instances of novel attributes such as synthesis of a novel view or expression. The methodology discussed above should be applicable to other deformable objects such as the human body and robots considering the kinematics of objects.

From my understanding, a PhD dissertation in science and engineering normally explains publishable new scientific results. New is to be respectful to the existing results; publishable is to ensure the quality of work. The author and his/her advisor and committee jointly take this responsibility. According to my knowledge, scientific results in computer science are mostly about methods. A method is definitely targeted at a certain problem. I need to make sure that it is a true problem - not a fabricated one. Doing projects and internships is to get myself exposed to true problems. Through repeated trial and error, I need to find out those methods that can actually work. I will develop the ability of problem solving through doing such

applied research - being able to solve a problem using a specific method and then even being able to solve more. However, I should not be confined to solving practical problems. Ideally I had better be able to abstract some general questions from those practical problems and abstract some generic methods from those specific methods. This process is normally achieved with the help of mathematical analysis. Sometimes I need to make some assumptions. I will have more stuffs to abstract from if Ive solved a few more practical problems through completing a few more projects. However, solving more practical problems itself cannot guarantee the abstraction. I think this capability of abstraction is one of the aims of doctoral education. It is also something that I hope to gradually learn from my advisor. Furthermore, as regards why a method is able to work, It is necessary to analyze the nature and properties of a method, which leads to a theoretical research. Ideally in the end, some PhD candidate can build a system oriented at a method, while some others might establish a theory oriented at contributing in understanding.

In my opinion, plenty of work can be done at each step of a research, either applied or theoretical. To produce something novel sometimes is harder. The field of computer science is actively changing. True and fabricated problems, working and not-working methods, come out one after another, increasing the difficulty of innovation. I will hardly find a problem that no one has ever tried to solve, design a method that no one has ever used, or propose an angle of understanding that no one has ever thought of. I have to keep understanding the problems that others

have recently discovered, the methods that others have recently designed and the perspective of understanding that others have recently proposed. Even if I cannot invent in the beginning, it is a good start to be able to evaluate whether a problem is a true valuable problem and whether a method is a working method. As Ive read a number of papers over years, I find the reality is that most of today's papers find a slightly different problem, design a slightly different approach, or propose a slightly different understanding. The reality is one thing, while the pursuit is another. If the pursuit is set to be as high as the reality, the reality would be even lower.